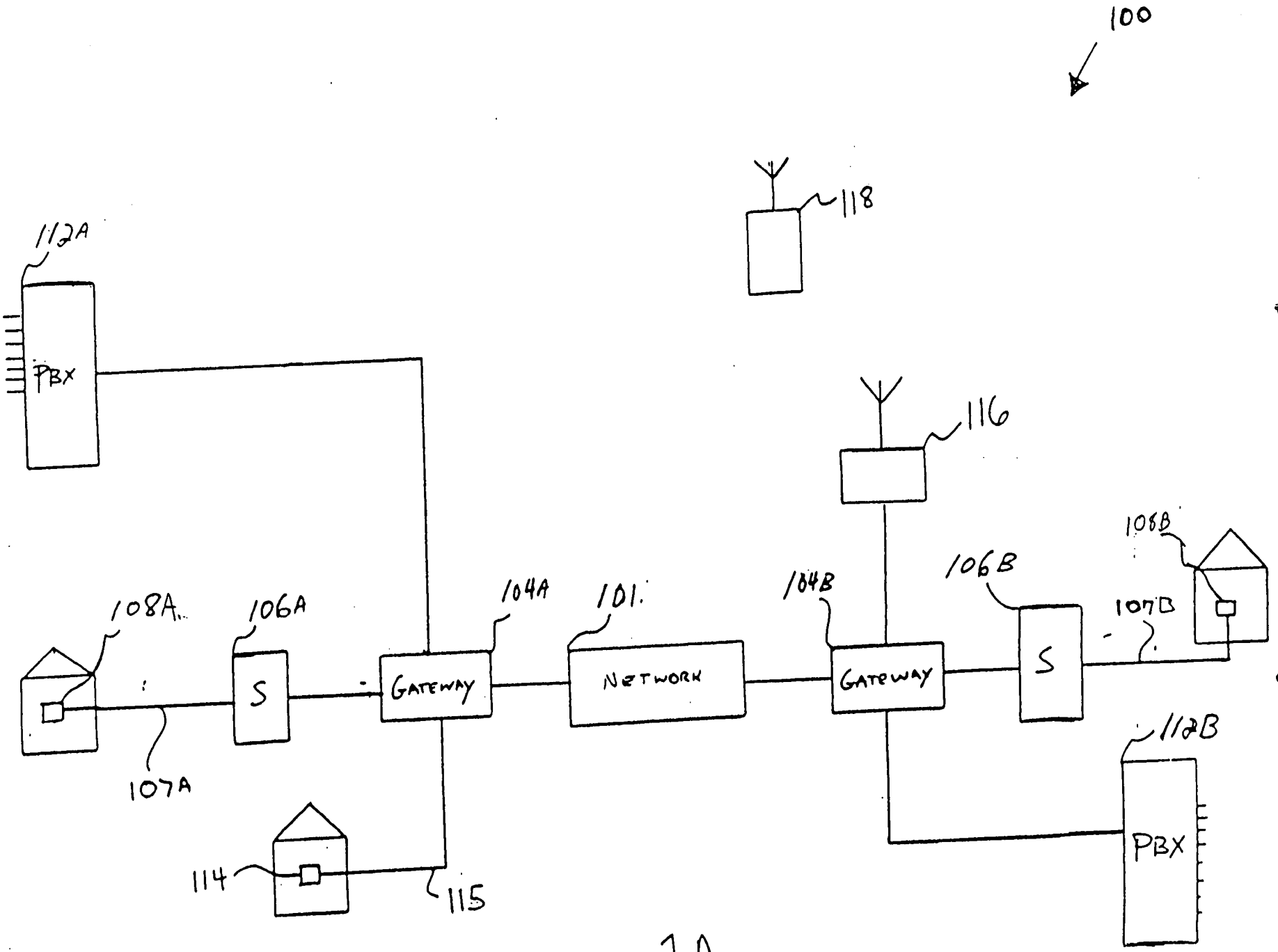

100
118
112A
PBX
116
108B
106B
108A
106A
104A
101
104B
107B
S
GATEWAY
NETWORK
GATEWAY
S
107A
112B
114
115
PBX

FIG. 1A

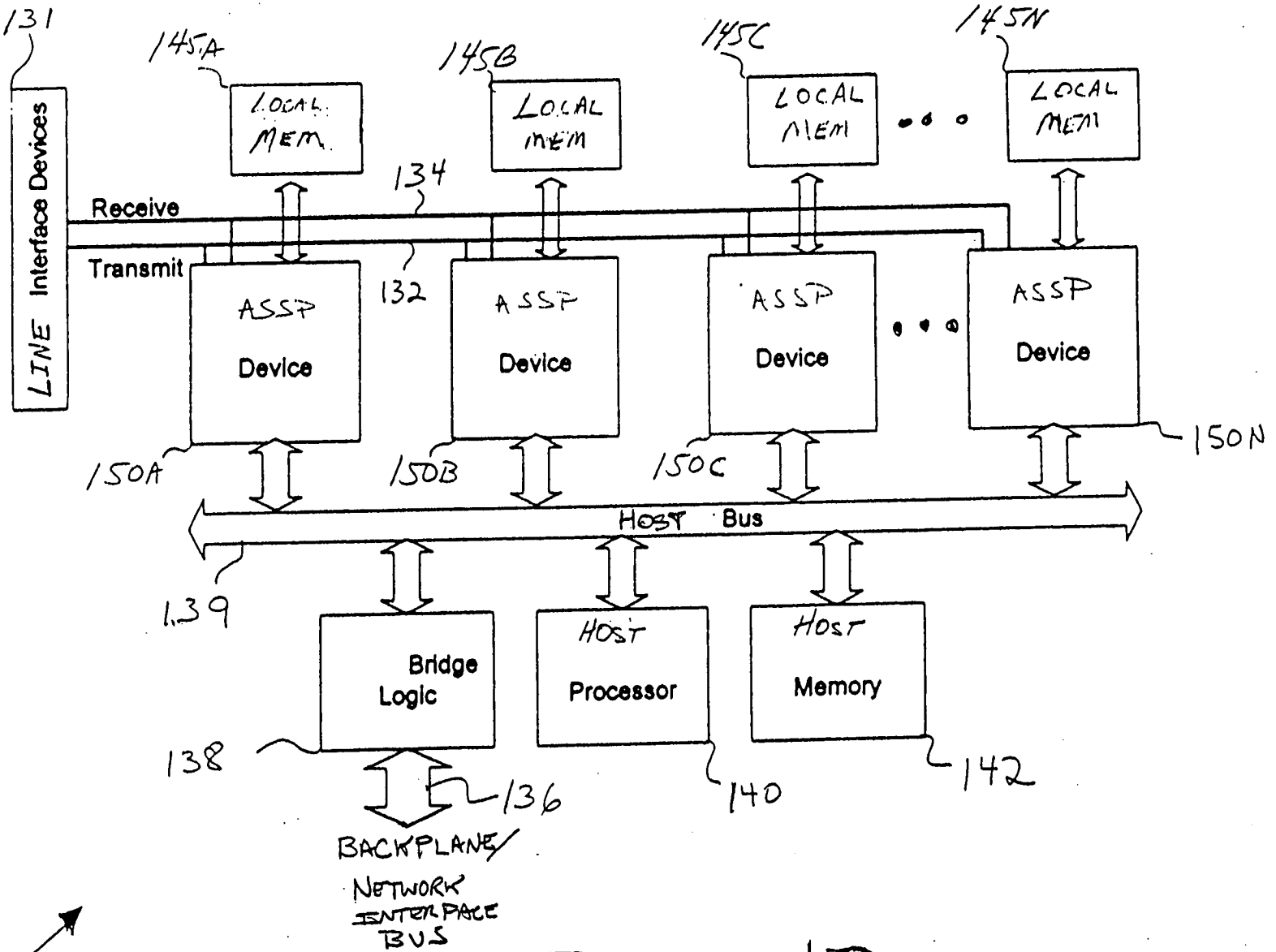

131
LINE Interface Devices
145A
LOCAL MEM.
145B
LOCAL MEM
145C
LOCAL MEM
145N
LOCAL MEM
134
Receive
Transmit
132
ASSP Device
ASSP Device
ASSP Device
ASSP Device
150A
150B
150C
150N
Host Bus
139
Bridge Logic
Host Processor
Host Memory
138
136
140
142
BACKPLANE/ NETWORK INTERFACE BUS
130

FIG. 1B

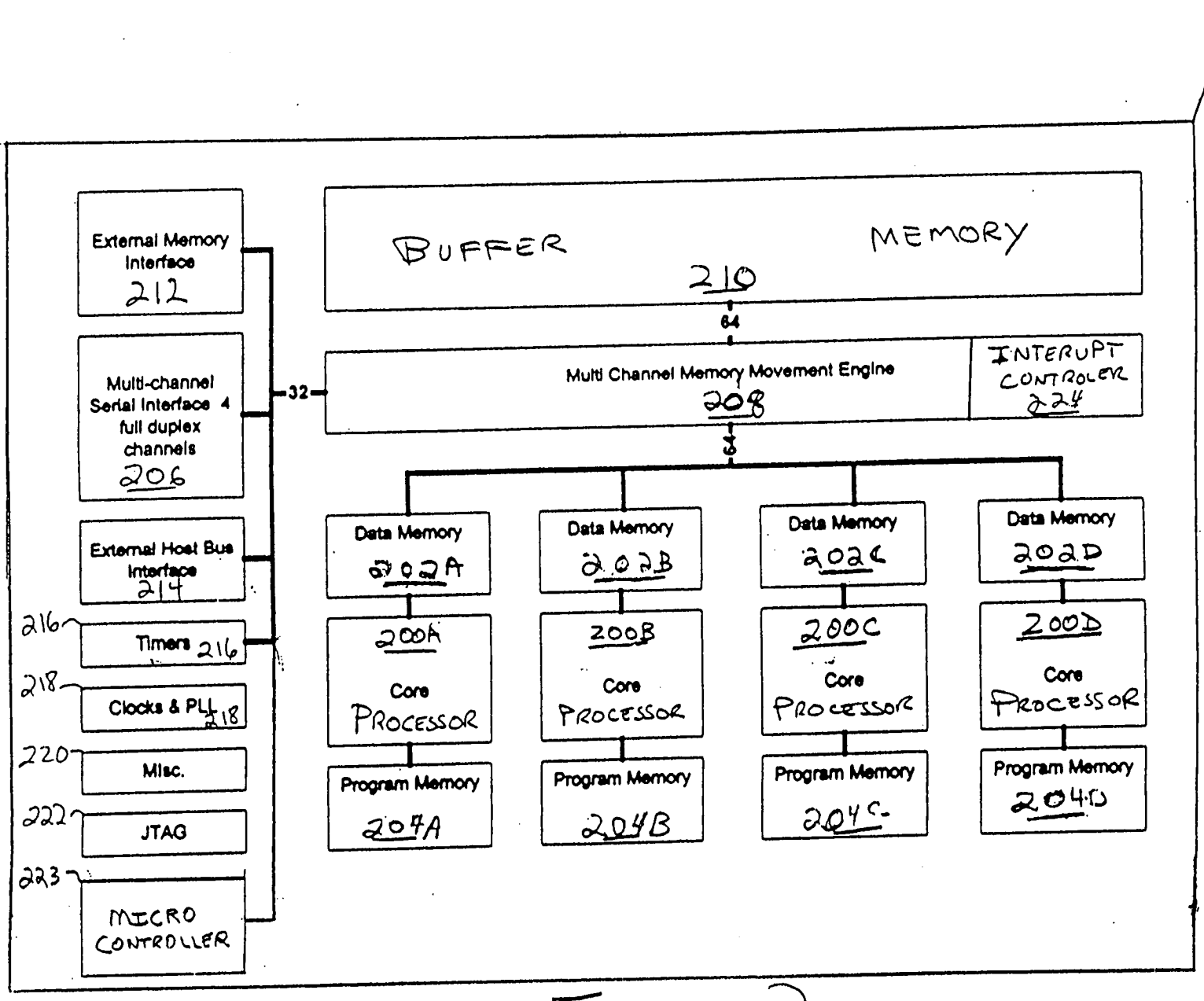

150
External Memory Interface 212
Multi-channel Serial Interface 4 full duplex channels 206
External Host Bus Interface 214
216
Timers 216
218
Clocks & PLL 218
220
Misc.
222
JTAG
223
MICRO CONTROLLER
32
BUFFER MEMORY 210
64
Multi Channel Memory Movement Engine 208
INTERUPT CONTROLER 224
Data Memory 202A
Data Memory 202B
Data Memory 202C
Data Memory 202D
200A Core PROCESSOR
200B Core PROCESSOR
200C Core PROCESSOR
200D Core PROCESSOR
Program Memory 204A
Program Memory 204B
Program Memory 204C
Program Memory 204D

FIG. 2

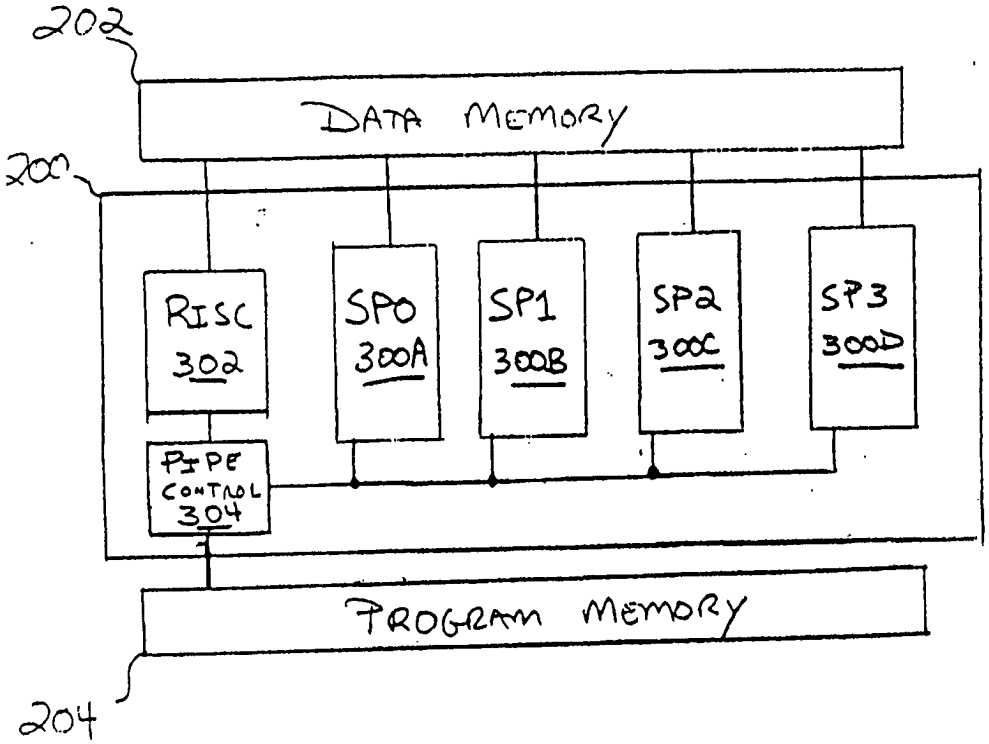



FIG. 3

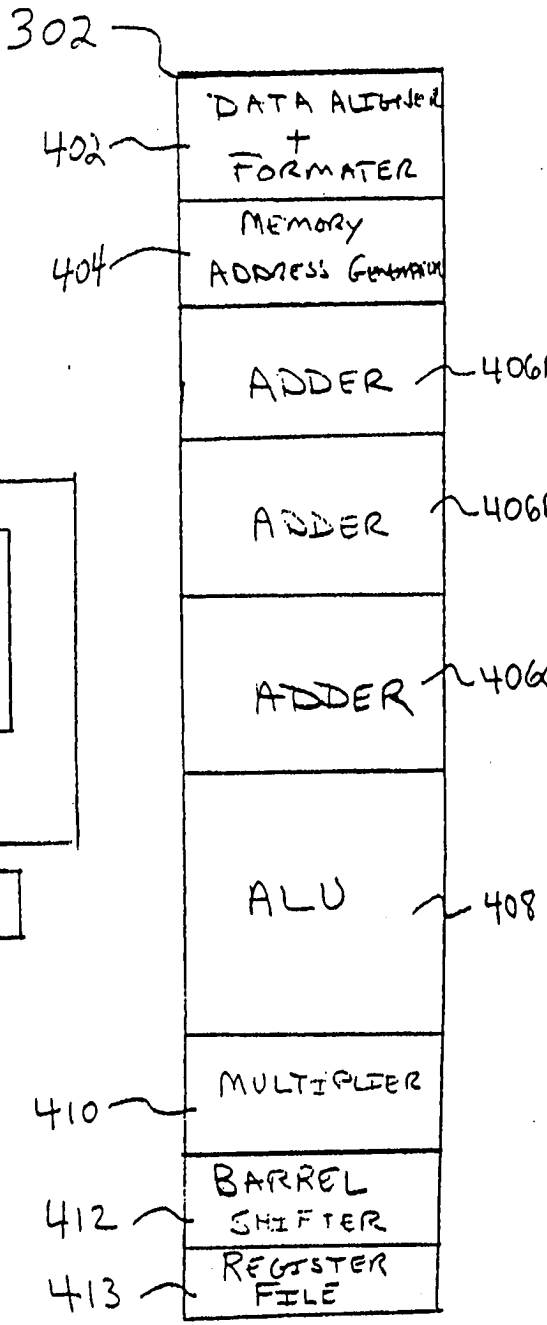



FIG. 4

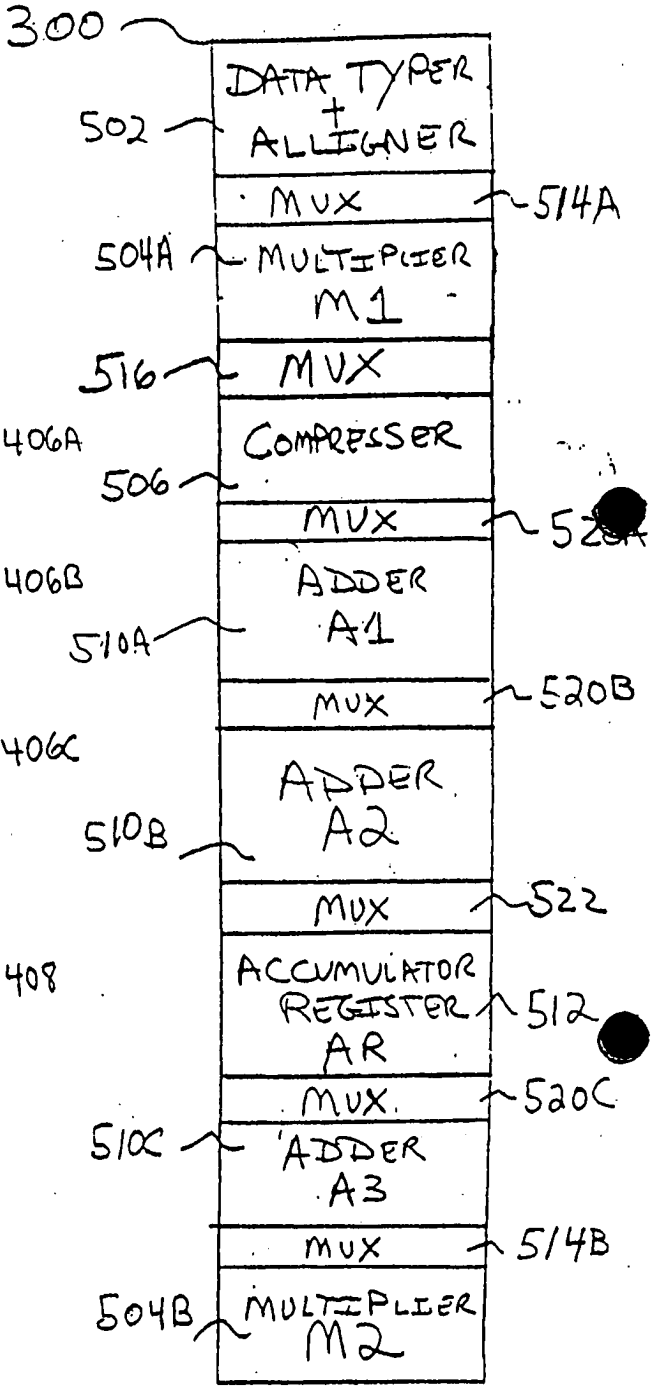



FIG. 5A

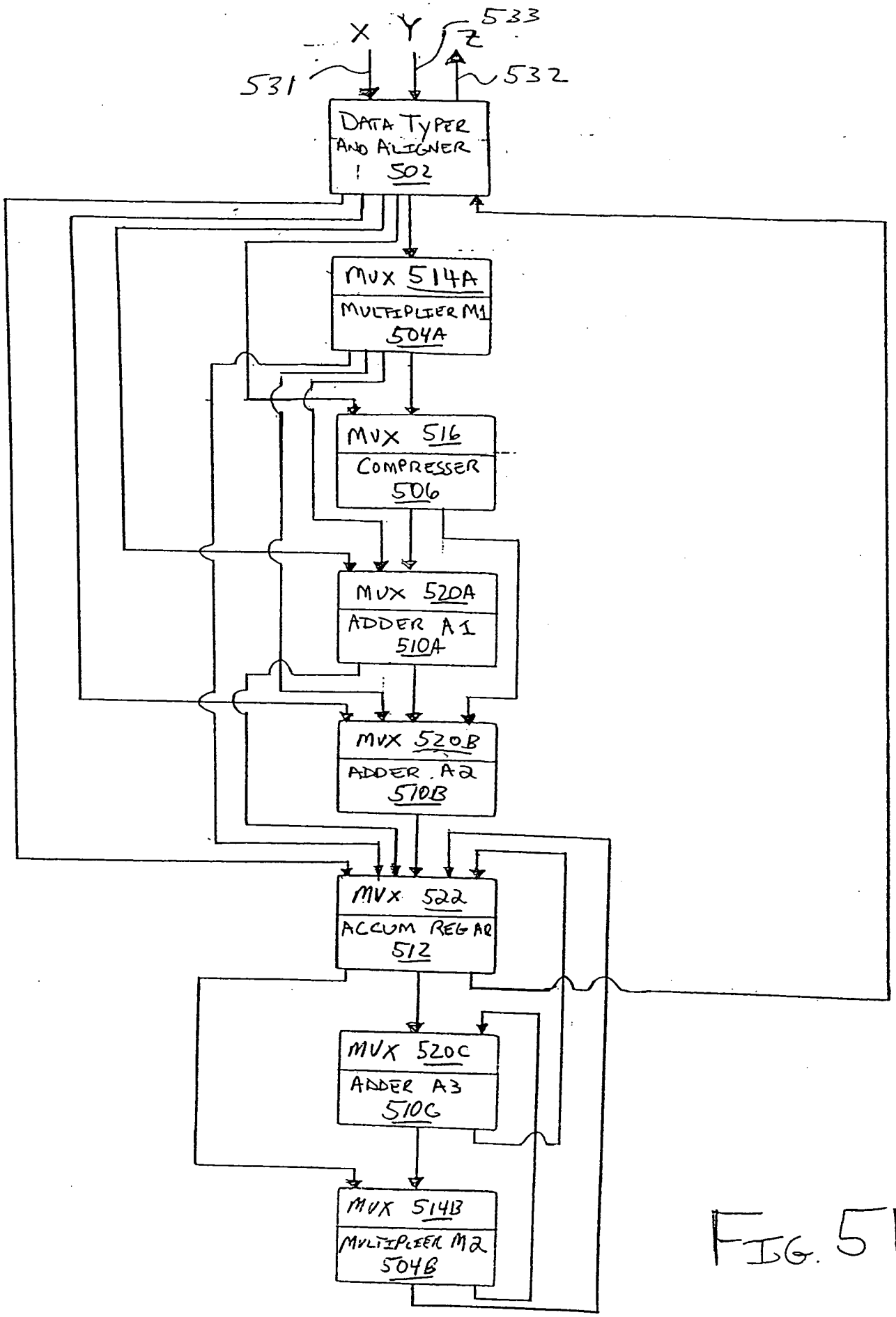

X
Y
Z
533
531
532
DATA TYPER AND ALIGNER 502
MUX 514A
MULTIPLIER M1 504A
MUX 516
COMPRESSER 506
MUX 520A
ADDER A1 510A
MUX 520B
ADDER A2 510B
MUX 522
ACCUM REG AR 512
MUX 520C
ADDER A3 510C
MUX 514B
MULTIPLIER M2 504B

FIG. 5E

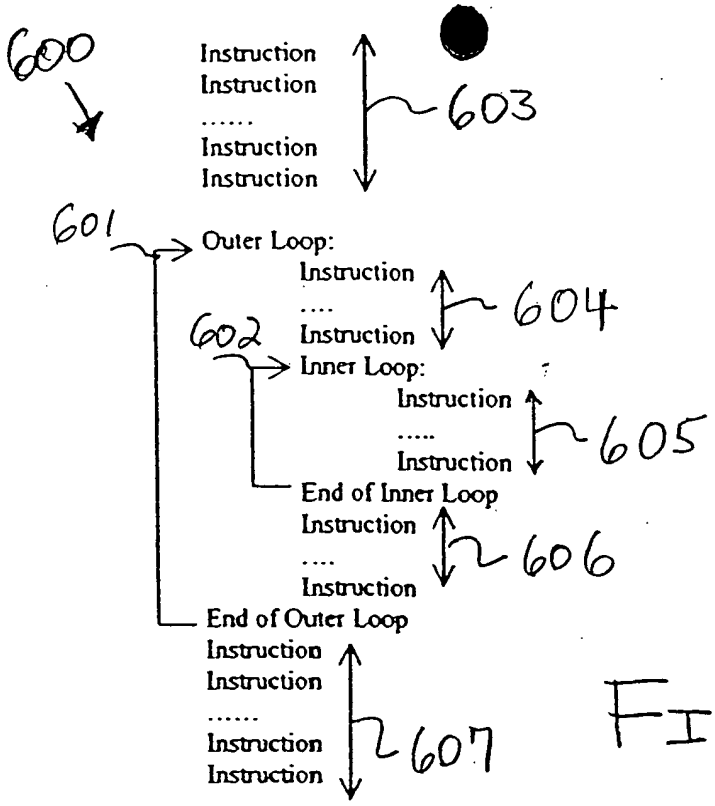



FIG. 6A

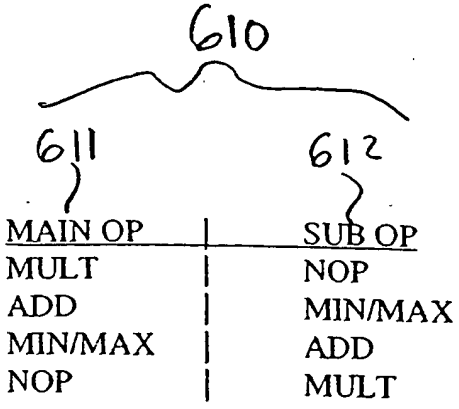



FIG. 6B

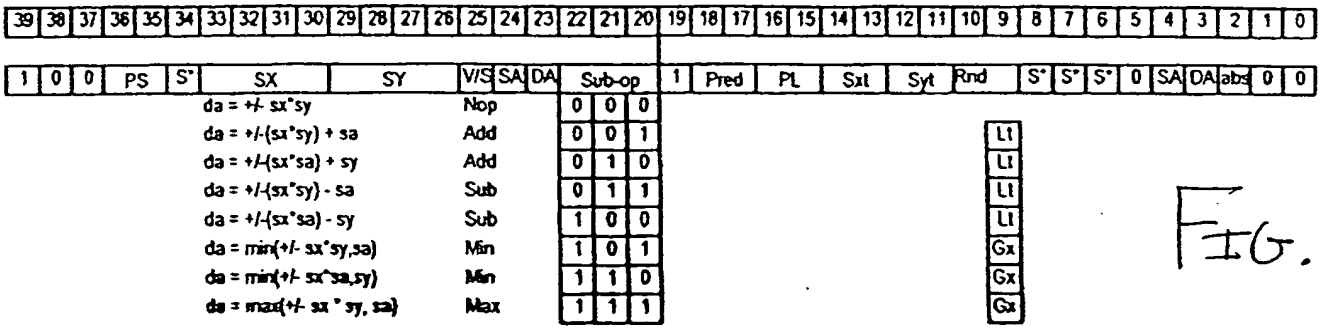



FIG. 6C

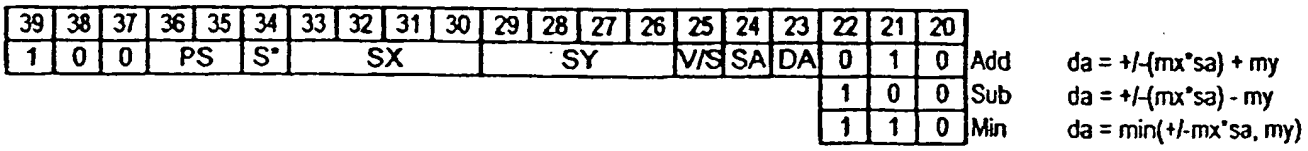



FIG. 6D

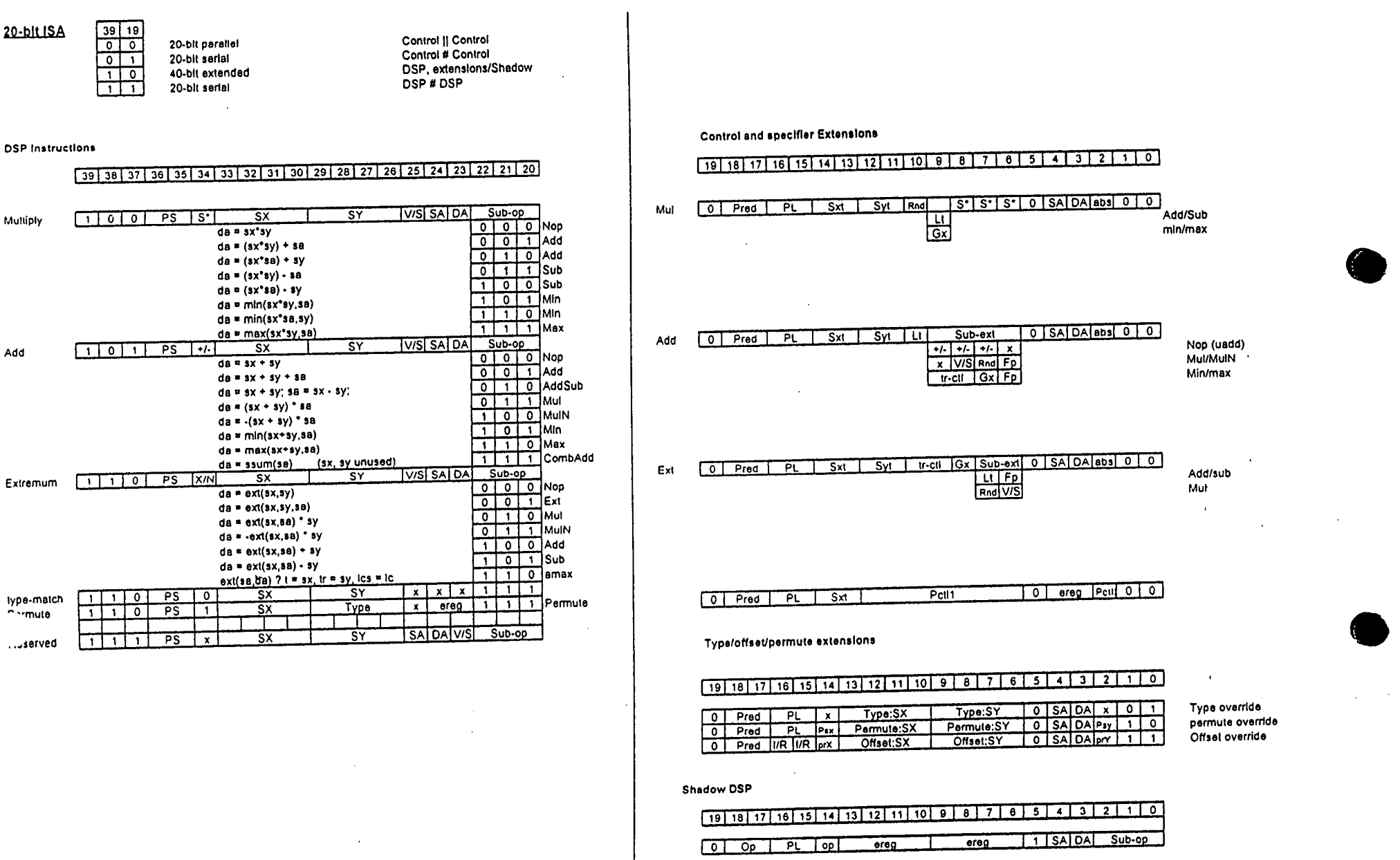

20-bit ISA
39 19
0 0 20-bit parallel
0 1 20-bit serial
1 0 40-bit extended
1 1 20-bit serial
Control || Control
Control # Control
DSP, extensions/Shadow
DSP # DSP
DSP Instructions
39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20
Multiply
1 0 0 PS S* SX SY V/S SA DA Sub-op
da = sx*sy
da = (sx*sy) + sa
da = (sx*sa) + sy
da = (sx*sy) - sa
da = (sx*sa) - sy
da = min(sx*sy,sa)
da = min(sx*sa,sy)
da = max(sx*sy,sa)
0 0 0 Nop
0 0 1 Add
0 1 0 Add
0 1 1 Sub
1 0 0 Sub
1 0 1 Min
1 1 0 Min
1 1 1 Max
Add
1 0 1 PS +/- SX SY V/S SA DA Sub-op
da = sx + sy
da = sx + sy + sa
da = sx + sy; sa = sx - sy;
da = (sx + sy) * sa
da = -(sx + sy) * sa
da = min(sx+sy,sa)
da = max(sx+sy,sa)
da = ssum(sa) (sx, sy unused)
0 0 0 Nop
0 0 1 Add
0 1 0 AddSub
0 1 1 Mul
1 0 0 MulN
1 0 1 Min
1 1 0 Max
1 1 1 CombAdd
Extremum
1 1 0 PS X/N SX SY V/S SA DA Sub-op
da = ext(sx,sy)
da = ext(sx,sy,sa)
da = ext(sx,sa) * sy
da = -ext(sx,sa) * sy
da = ext(sx,sa) + sy
da = ext(sx,sa) - sy
ext(sa,ba) ? t = sx, tr = sy, lcs = lc
0 0 0 Nop
0 0 1 Ext
0 1 0 Mul
0 1 1 MulN
1 0 0 Add
1 0 1 Sub
1 1 0 amax
Type-match Permute
1 1 0 PS 0 SX SY x x x 1 1 1 Permute
1 1 0 PS 1 SX Type x ereg 1 1 1
Reserved
1 1 1 PS x SX SY SA DA V/S Sub-op
Control and specifier Extensions
19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Mul
0 Pred PL Sxt Syt Rnd S* S* S* 0 SA DA abs 0 0
Lt
Gx
Add/Sub
min/max
Add
0 Pred PL Sxt Syt Lt Sub-ext 0 SA DA abs 0 0
+/- +/- +/- x
x V/S Rnd Fp
tr-ctl Gx Fp
Nop (uadd)
Mul/MulN
Min/max
Ext
0 Pred PL Sxt Syt tr-ctl Gx Sub-ext 0 SA DA abs 0 0
Lt Fp
Rnd V/S
Add/sub
Mul
0 Pred PL Sxt Pctl1 0 ereg Pctl 0 0
Type/offset/permute extensions
19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0 Pred PL x Type:SX Type:SY 0 SA DA x 0 1
0 Pred PL Psx Permute:SX Permute:SY 0 SA DA Psy 1 0
0 Pred I/R I/R prx Offset:SX Offset:SY 0 SA DA prY 1 1
Type override
permute override
Offset override
Shadow DSP
19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
0 Op PL op ereg ereg 1 SA DA Sub-op

FIG. 6 E

**Control Instructions**

| | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| add,sub | L | Pred | | 0 | 0 | 0 | RX | | | | RY | | | | RZ | | | | +/- | 0 |
| max,min | L | Pred | | 0 | 0 | 0 | RX | | | | RY | | | | RZ | | | | X/N | 1 |
| Shift | L | Pred | | 0 | 0 | 1 | RX | | | | UI4 | | | | RZ | | | | UI1 | R/L |
| Logic | L | Pred | | 0 | 1 | 0 | RX | | | | RY | | | | RZ | | | | &, \|, &\| | |
| Mux | L | Pred | | 0 | 1 | 1 | RX | | | | RY | | | | RZ | | | | Pd | 0 |
| mov | L | Pred | | 0 | 1 | 1 | RX | | | | DZ | | | | Rxt | Dzt | 0 | 0 | 0 | 1 |
| addi | L | Pred | | 0 | 1 | 1 | SI4 | | | | DZ | | | | x | x | 1 | 0 | 0 | 1 |
| mov2erg | L | Pred | | 0 | 1 | 1 | RX | | | | unit | | ereg | | gd | | type | | 1 | 0 | 1 |
| | L | Pred | | 0 | 1 | 1 | RX | | | | DZ1 | | | | DZ2 | | | | 1 | 1 |
| bits | L | Pred | | 1 | 0 | 0 | UI4:POS | | | | RZ | | | | Rzt | UI4 | | | | 0 |
| set2bits | L | Pred | | 1 | 0 | 0 | UI4:POS | | | | RZ | | | | Rzt | UI2 | | 0 | 0 | 1 |
| Setbit | L | Pred | | 1 | 0 | 0 | UI4:POS | | | | RZ | | | | Rzt | UI1 | UI1 | 1 | 0 | 1 |
| Movi | L | Pred | | 1 | 0 | 0 | SI8 | | | | | | | | RZ | | | | 1 | 1 |
| Jmp | L | Pred | | 1 | 0 | 1 | SI9 | | | | | | | | 0 | Pred | | | 0 | 0 |
| Call | L | Pred | | 1 | 0 | 1 | SI9 | | | | | | | | 1 | Pred | | | 0 | 0 |
| Loop | L | Pred | | 1 | 0 | 1 | UI5 : Lcount | | | | UI5: Lsize | | | | | | | UI2: Lst | 0 | 1 |
| Jmpi | L | Pred | | 1 | 0 | 1 | RX | | | | x | x | x | x | x | 0 | Pred | | 1 | 0 |
| Calli | L | Pred | | 1 | 0 | 1 | RX | | | | x | x | x | x | x | 1 | Pred | | 1 | 0 |
| Loopi | L | Pred | | 1 | 0 | 1 | RX | | | | x | UI5; Lsize | | | | | | UI2: Lst | 1 | 1 |
| Test | L | Pred | | 1 | 1 | 0 | RX | | | | RY | | | | PZ | | | | =, <, > | 0 |
| Testbit | L | Pred | | 1 | 1 | 0 | RX | | | | UI5 | | | | PZ | | | B | 0 | 1 |
| Andp, orp | L | Pred | | 1 | 1 | 0 | Pa | | | | Pb | | Pc | | PZ | | | &\| | 1 | 1 |
| Load | L | Pred | | 1 | 1 | 1 | MX | | | | RZ | | | | Ext | | | | 0 | 0 | 0 |
| Store | L | Pred | | 1 | 1 | 1 | MZ | | | | RX | | | | Ext | | | | 1 | 0 | 0 |
| eLoad | L | Pred | | 1 | 1 | 1 | MX | | | | RZ | | | | 1 | 1 | 1 | 0 | 0 | 0 |
| eStore | L | Pred | | 1 | 1 | 1 | MZ | | | | RX | | | | 1 | 1 | 1 | 1 | 0 | 0 |
| Extended | L | Pred | | 1 | 1 | 1 | Bits 27:16 | | | | | | | | | | | | 1 | 0 |
| Logic2 | L | Pred | | 1 | 1 | 1 | RX | | | | RY/RZ | | | | Rxt | Ryt | &, \|, &\|, ! | | 0 | 1 |
| mov-erg | L | Pred | | 1 | 1 | 1 | unit | | ereg | | RZ | | | | gd | | Sft | | 0 | 1 | 1 |
| Crb | L | Pred | | 1 | 1 | 1 | RX | | | | RZ | | | | s/m | 0 | 0 | 1 | 1 | 1 |
| Parity | L | Pred | | 1 | 1 | 1 | RX | | | | PZ | | | O/E | 0 | 1 | 0 | 1 | 1 | 1 |
| Stm | L | Pred | | 1 | 1 | 1 | MZ | | | | RX | | | | 1 | 1 | 0 | 1 | 1 | 1 |
| Abs | L | Pred | | 1 | 1 | 1 | RX | | | | RZ | | | | 0 | 0 | 1 | 1 | 1 | 1 |
| Neg | L | Pred | | 1 | 1 | 1 | RX | | | | RZ | | | | 0 | 1 | 1 | 1 | 1 | 1 |
| step | L | Pred | | 1 | 1 | 1 | RX | | | | RZ | | | | 1 | 0 | 1 | 1 | 1 | 1 |
| t & Set | L | Pred | | 1 | 1 | 1 | RX | | | | PZ | | | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| Reserved | L | Pred | | 1 | 1 | 1 | Pred | | I-ctl | | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Return | L | Pred | | 1 | 1 | 1 | ac # | | | | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Zero-ac | L | Pred | | 1 | 1 | 1 | RZ | | | | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| eSync | L | Pred | | 1 | 1 | 1 | UI3 | | | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Swi | L | Pred | | 1 | 1 | 1 | UI3 | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Nop | L | Pred | | 1 | 1 | 1 | UI3 | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

<Bit1, Bits9-6> == UI5 (Shift Amount)

<Bit3, Bits13-10> == UI5 :POS

FIG. 6F

**Extended Control**

| | Bits 13:2 of upper half (39:20) | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Insert/Extract | RX | | | | RZ | | | | 0 | 0 | 0 | 0 | 0 | x | x | 0 | Rxt | Rzt | I/E | R/I | R/I | Offset:UI5 | | | | | Length:UI5 | | | | | 0 |
| | | | | | | | | | | | | | | | | | | | | | | x | RY | | | | RV | | | | x | |
| Inserti | UI4: length | | | | RZ | | | | 0 | 0 | 0 | 1 | 0 | x | x | 0 | Rzt | UI5: Position | | | | Imm10 | | | | | | | | | | |
| Shift | RX | | | | RZ | | | | 0 | 0 | 0 | 0 | 0 | rxh | rxl | 0 | rzh | rzl | D | U/S | 1 | Shift: UI5 | | | | | A/L | Lt | R/L | 0 | Fill | 1 |
| | | | | | | | | | | | | | | | | | | | | | 0 | ryh | RY | | | | | | | | | |
| Rotate | RX | | | | RZ | | | | 0 | 0 | 0 | 0 | 0 | x | x | 0 | x | x | x | x | 1 | Shift: UI5 | | | | | 1 | x | R/L | 1 | x | 1 |
| | | | | | | | | | | | | | | | | | | | | | 0 | ryh | RY | | | | | | | | | |
| Jmp, call | UI7 | | | | | | | J/C | 0 | 0 | 1 | 0 | 0 | Pred | | 0 | x | UI15 | | | | | | | | | | | | | | |
| dloop | UI4: outer LC | | | | UI4: inner LC | | | | 0 | 0 | 1 | 1 | 0 | x | exit | 0 | UI1 | UI4: outer Lsize | | | | UI4: inner Lsize | | | UI2: O-Ls | | UI4: inner Lstart | | | | 0 |
| pi | RX | | | | RY | | | | 0 | 0 | 1 | 1 | 0 | x | exit | 0 | x | UI4: outer Lsize | | | | UI4: inner Lsize | | | UI2: O-Ls | | UI4: inner Lstart | | | | 1 |
| | RX | | | | RY | | | | 0 | 1 | 0 | 0 | 0 | x | x | 0 | 0 | rxh | rxl | ryh | +/- | =/+ | RZ | | | | | I/f | rzh | rzl | s/u | s/u | 0 |
| add/sub | RX | | | | RY | | | | 0 | 1 | 0 | 0 | 0 | x | x | 0 | 0 | rxh | rxl | ryh | ryl | +/- | RZ | | | | | Lt | rzh | rzl | x | x | 1 |
| Reserved | RX | | | | RY | | | | 0 | 1 | 0 | 0 | 0 | x | x | 0 | 0 | T/F | T/F | T/F | &\| | &\| | PY | | | | | PV | | | | x | 1 |
| logicp | PX | | | D | PZ | | | | 0 | 1 | 0 | 0 | 0 | x | x | 0 | 1 | T/F | T/F | T/F | &\| | &\| | PY | | | | | PV | | | | x | 1 |
| Testi | RX | | | D | PZ | | | | 0 | 1 | 0 | 1 | 0 | =,>,< | | 0 | Imm16 | | | | | | | | | | | | | | | |
| Movl | H/L | Fill | | | RZ | | | | 0 | 1 | 1 | 0 | 0 | x | x | 0 | Imm16 | | | | | | | | | | | | | | | |
| loadi | Type | | | | RZ | | | | 0 | 1 | 1 | 1 | 0 | x | x | 0 | 0 | 0 | Imm14 | | | | | | | | | | | | | | |
| storei | Type | | | | RX | | | | 0 | 1 | 1 | 1 | 0 | x | x | 0 | 0 | 1 | Imm14 | | | | | | | | | | | | | | |
| loadt | RX | | | | RZ | | | | 0 | 1 | 1 | 1 | 0 | x | x | 0 | 1 | Rzt | 0 | Type | | SI10 | | | | | | | | | | |
| storet | MZ | | | | RX | | | | 0 | 1 | 1 | 1 | 0 | x | x | 0 | 1 | Rzt | 1 | Type | | SI10 | | | | | | | | | | |
| Addi/subi | RX | | | | RZ | | | | 1 | 0 | +/- | 0 | 0 | Lt | s/u | 0 | Imm16 | | | | | | | | | | | | | | | |
| mini,maxi | RX | | | | RZ | | | | 1 | 0 | X/N | 1 | 0 | x | x | 0 | Imm16 | | | | | | | | | | | | | | | |
| andi,ori | RX | | | | RZ | | | | 1 | 1 | &\| | H/L | 0 | x | x | 0 | Imm16 | | | | | | | | | | | | | | | |

Fill: Sign/Zero

Bit 15 is continuation of Inner LC

andp, orp, andorp, orandp: pz = (px relop py) relop pv)

FIG. 6G

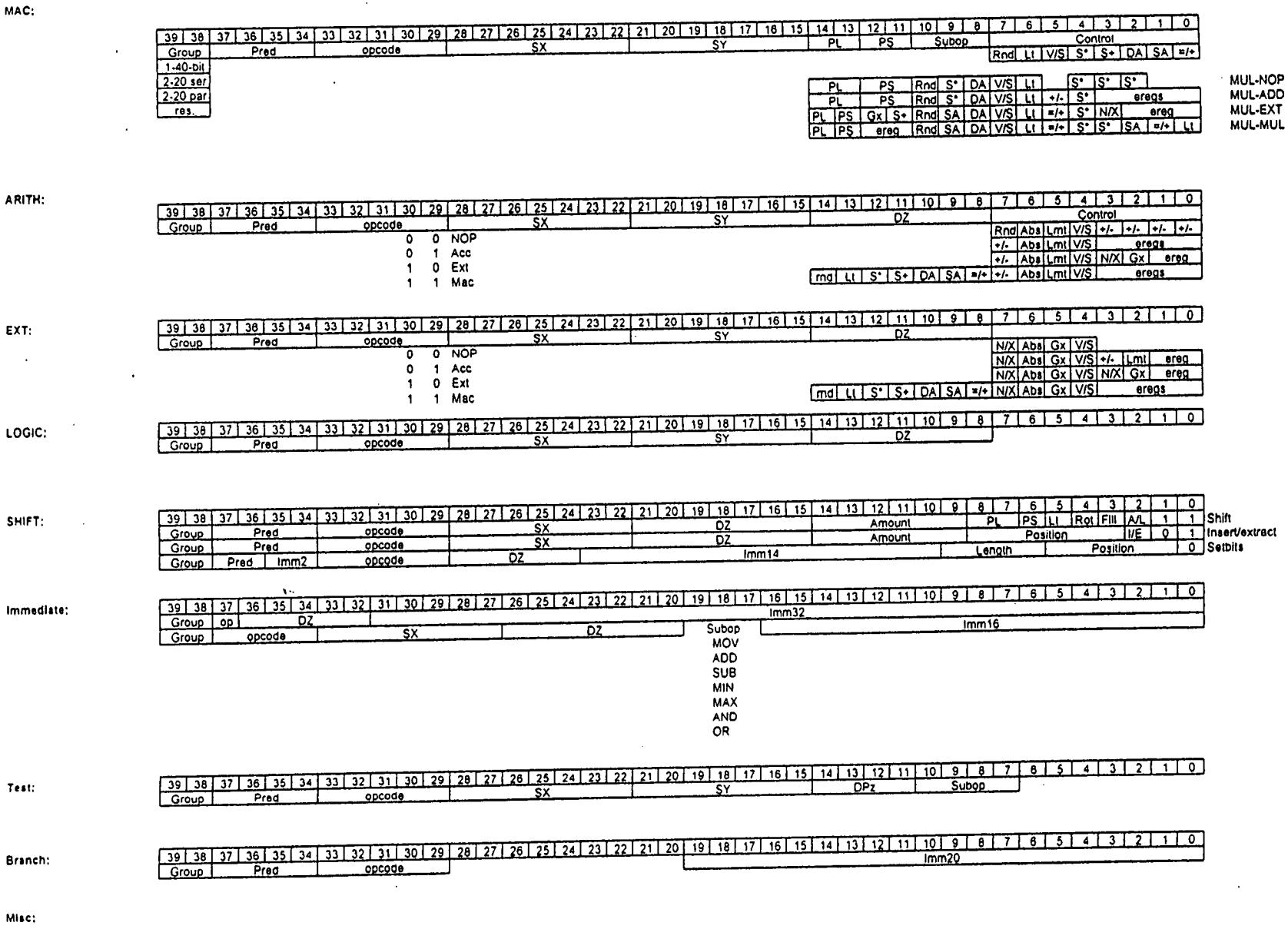
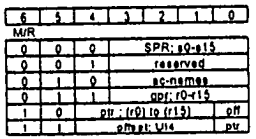
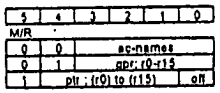

MAC:
Group
Pred
opcode
SX
SY
PL
PS
Subop
Control
Rnd
Lt
V/S
S*
S+
DA
SA
=/+
1-40-bit
2-20 ser
2-20 par
res.
MUL-NOP
MUL-ADD
MUL-EXT
MUL-MUL
ARITH:
DZ
Abs
Lmt
+/-
eregs
ereg
Gx
N/X
0 0 NOP
0 1 Acc
1 0 Ext
1 1 Mac
EXT:
LOGIC:
SHIFT:
Amount
Position
Length
Imm14
Imm2
Shift
Insert/extract
Setbits
Immediate:
op
Imm32
Imm16
Subop
MOV
ADD
SUB
MIN
MAX
AND
OR
Test:
DPz
Branch:
Imm20
Misc:

FIG. 6H

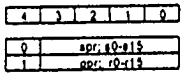
7-bit specifier: Parallel Store, Parallel Load in DSP instructions



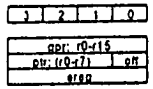

6-bit specifier: DSP instructions



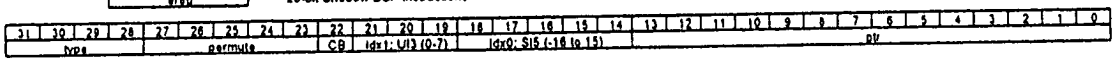

specifier: RISC instructions




4-bit specifier:




AR:




ac-names:

| 3 | 2 | 1 | 0 | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | A0 (use type, SIMD) |
| 0 | 0 | 0 | 1 | A1 |
| 0 | 0 | 1 | 0 | T |
| 0 | 0 | 1 | 1 | TR |
| 0 | 1 | 0 | 0 | A00 (unit 0) |
| 0 | 1 | 0 | 1 | A10 |
| 0 | 1 | 1 | 0 | T0 |
| 0 | 1 | 1 | 1 | TR0 |
| 1 | 0 | 0 | 0 | SX1 |
| 1 | 0 | 0 | 1 | SX1e |
| 1 | 0 | 1 | 0 | SX2 |
| 1 | 0 | 1 | 1 | SX2e |
| 1 | 1 | 0 | 0 | SY1 |
| 1 | 1 | 0 | 1 | SY1e |
| 1 | 1 | 1 | 0 | SY2 |
| 1 | 1 | 1 | 1 | SY2e |

SPR:

gpr-type
areg-type
fu-ctl
pte-ctl
cb-ctl
loop-ctl
pcr
status

areg-names:

| 3 | 2 | 1 | 0 | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | A0 |
| 0 | 0 | 0 | 1 | A1 |
| 0 | 0 | 1 | 0 | T |
| 0 | 0 | 1 | 1 | TR |
| 0 | 1 | 0 | 0 | PP0 |
| 0 | 1 | 0 | 1 | Aout |
| 0 | 1 | 1 | 0 | PP1 |
| 0 | 1 | 1 | 1 | Dout |
| 1 | 0 | 0 | 0 | SX1 |
| 1 | 0 | 0 | 1 | SX1e |
| 1 | 0 | 1 | 0 | SX2 |
| 1 | 0 | 1 | 1 | SX2e |
| 1 | 1 | 0 | 0 | SY1 |
| 1 | 1 | 0 | 1 | SY1e |
| 1 | 1 | 1 | 0 | SY2 |
| 1 | 1 | 1 | 1 | SY2e |

FIG. 6I

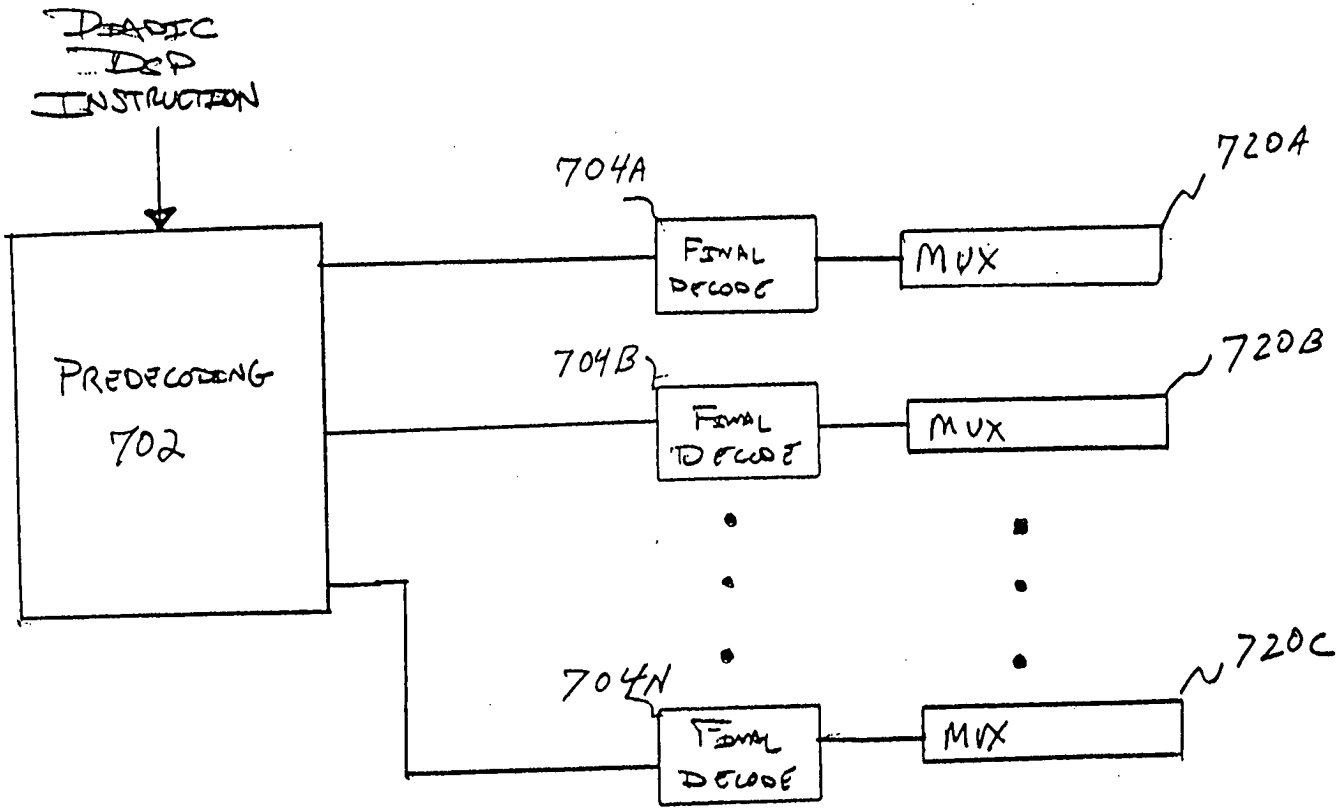

DYADIC DSP INSTRUCTION
PREDECODING 702
704A
FINAL DECODE
MUX
720A
704B
FINAL DECODE
MUX
720B
704N
FINAL DECODE
MUX
720C

FIG. 7

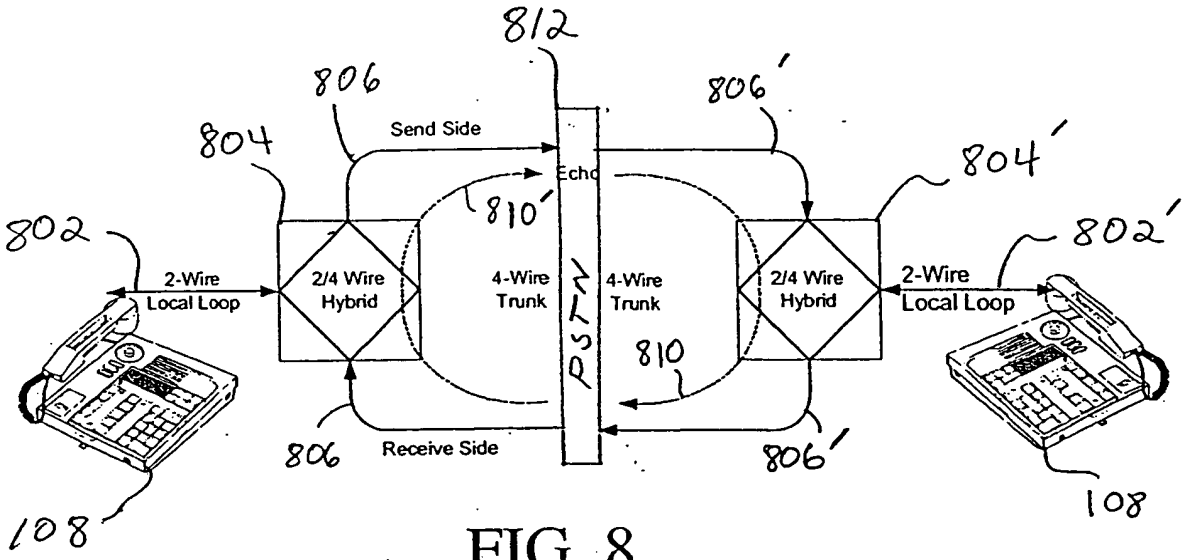



FIG. 8
(PRIOR ART)

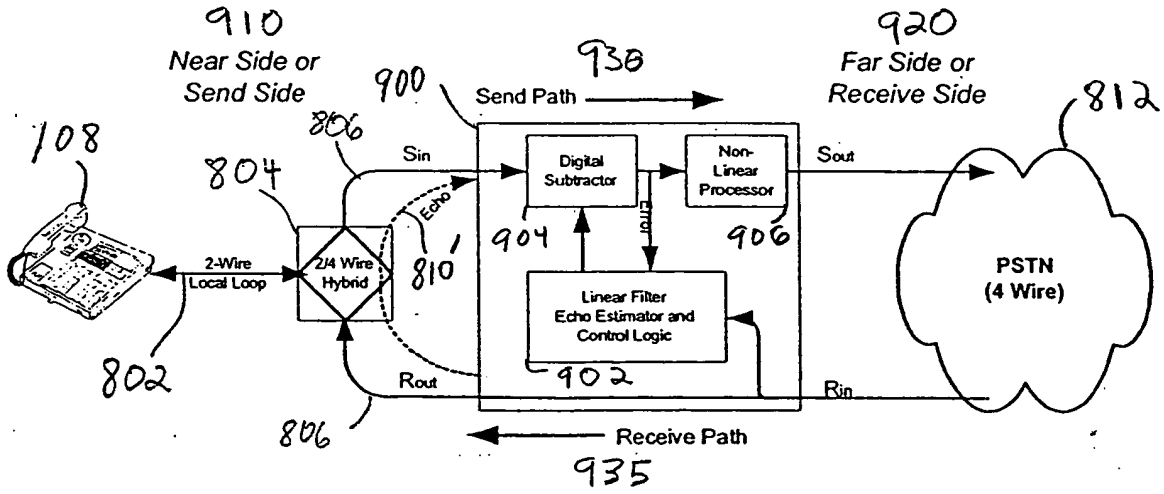



FIG. 9
(PRIOR ART)

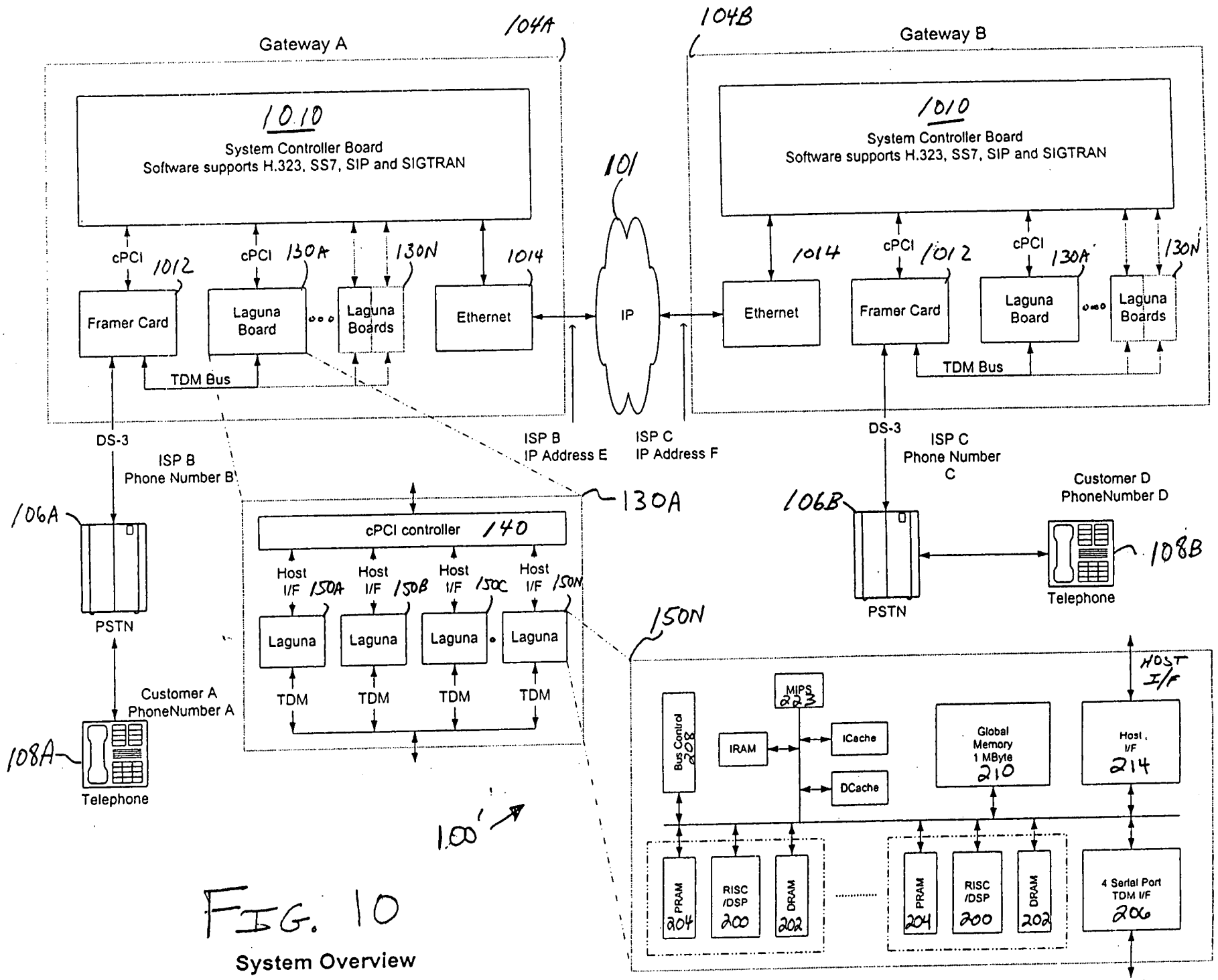



FIG. 10

System Overview

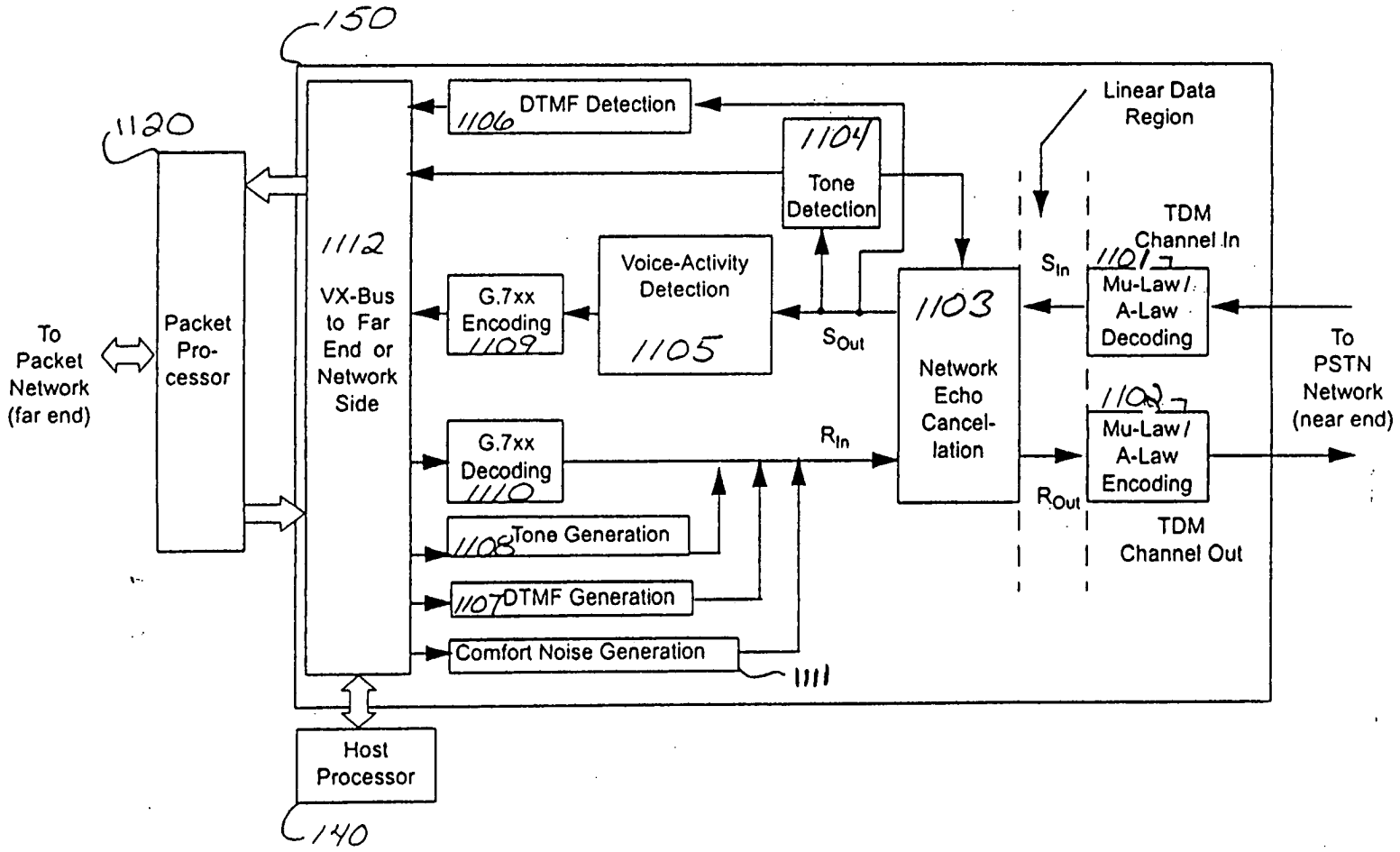

150
1120
1112
VX-Bus to Far End or Network Side
Packet Pro-cessor
To Packet Network (far end)
Host Processor
140
DTMF Detection
1106
Tone Detection
1104
Voice-Activity Detection
1105
G.7xx Encoding
1109
G.7xx Decoding
1110
Tone Generation
1108
DTMF Generation
1107
Comfort Noise Generation
1111
Network Echo Cancel-lation
1103
Linear Data Region
TDM Channel In
Mu-Law / A-Law Decoding
Mu-Law / A-Law Encoding
TDM Channel Out
To PSTN Network (near end)
SIn
SOut
RIn
ROut

FIG. 11A

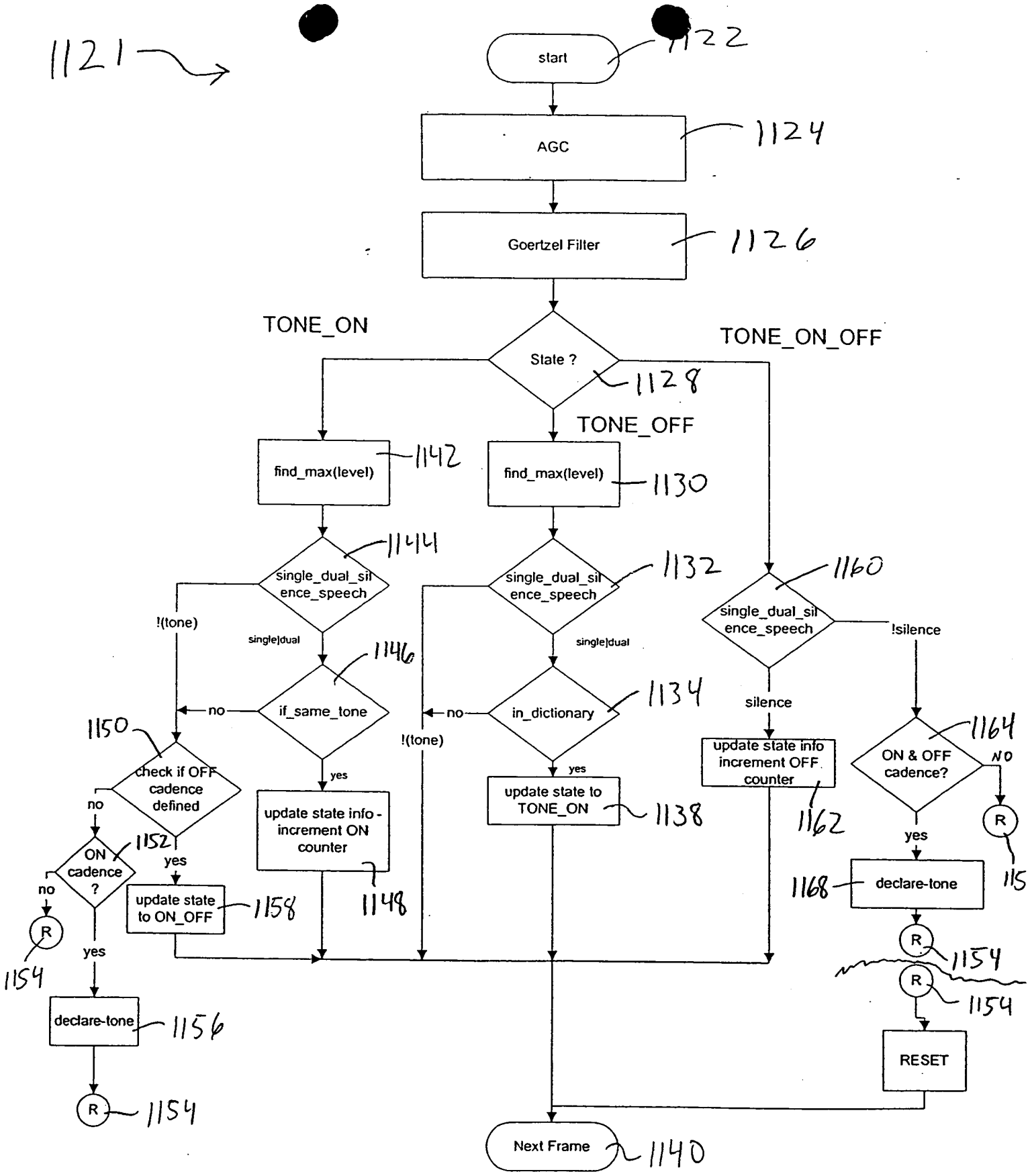

1121
start
1122
AGC
1124
Goertzel Filter
1126
TONE_ON
State ?
1128
TONE_ON_OFF
TONE_OFF
find_max(level)
1142
find_max(level)
1130
single_dual_silence_speech
1144
single_dual_silence_speech
1132
single_dual_silence_speech
1160
!(tone)
single|dual
1146
if_same_tone
no
yes
1150
check if OFF cadence defined
no
yes
1152
ON cadence ?
no
yes
update state to ON_OFF
1158
update state info - increment ON counter
1148
R
1154
declare-tone
1156
R
1154
single|dual
in_dictionary
1134
no
!(tone)
yes
update state to TONE_ON
1138
silence
update state info increment OFF counter
1162
!silence
ON & OFF cadence?
1164
NO
yes
R
115
1168
declare-tone
R
1154
R
1154
RESET
Next Frame
1140

FIG. 11B

Exemplary Filter coefficients for Goertzel Filter

| frequency | cos(2*pi*11/fs) | frequency index |
|---|---|---|
| 350 | 31536 | 0 |
| 400 | 31163 | 1 |
| 425 | 30958 | 2 |
| 440 | 30829 | 3 |
| 480 | 30465 | 4 |
| 540 | 29863 | 5 |
| 600 | 29195 | 6 |
| 620 | 28958 | 7 |
| 660 | 28462 | 8 |
| 697 | 27978 | 9 |
| 700 | 27938 | 10 |
| 770 | 26955 | 11 |
| 780 | 26808 | 12 |
| 852 | 25700 | 13 |
| 900 | 24916 | 14 |
| 941 | 24218 | 15 |
| 1020 | 22802 | 16 |
| 1100 | 21280 | 17 |
| 1140 | 20487 | 18 |
| 1209 | 19072 | 19 |
| 1300 | 17120 | 20 |
| 1336 | 16324 | 21 |
| 1380 | 15332 | 22 |
| 1477 | 13084 | 23 |
| 1500 | 12539 | 24 |
| 1620 | 9634 | 25 |
| 1633 | 9314 | 26 |
| 1700 | 7649 | 27 |
| 1740 | 6644 | 28 |
| 1860 | 3595 | 29 |
| 1980 | 514 | 30 |
| 2040 | -1029 | 31 |
| 2100 | -2570 | 32 |
| 2280 | -7147 | 33 |
| 2400 | -10125 | 34 |
| 2600 | -14875 | 35 |
| 3825 | -32457 | 36 |

FIG. 11C

Exemplary Call Progress Tones

| Frequency1 | Frequency2 | Call Progress Tone |
|---|---|---|
| 350 | 440 | ANSI T1.401 dial tone |
| 425 | 0 | Q.35 Dial Tone |
| 440 | 480 | ANSI T1.401 audible ringing |
| 480 | 620 | ANSI T1.401line busy tone |
| 480 | 620 | ANSI T1.401Reorder |
| 400 | 0 | Audible ringing |
| 440 | 0 | Dial Tone |
| 440 | 0 | ANSI T1.401Fast Busy Tone |
| 440 | 0 | Busy Tone |

FIG. 11D

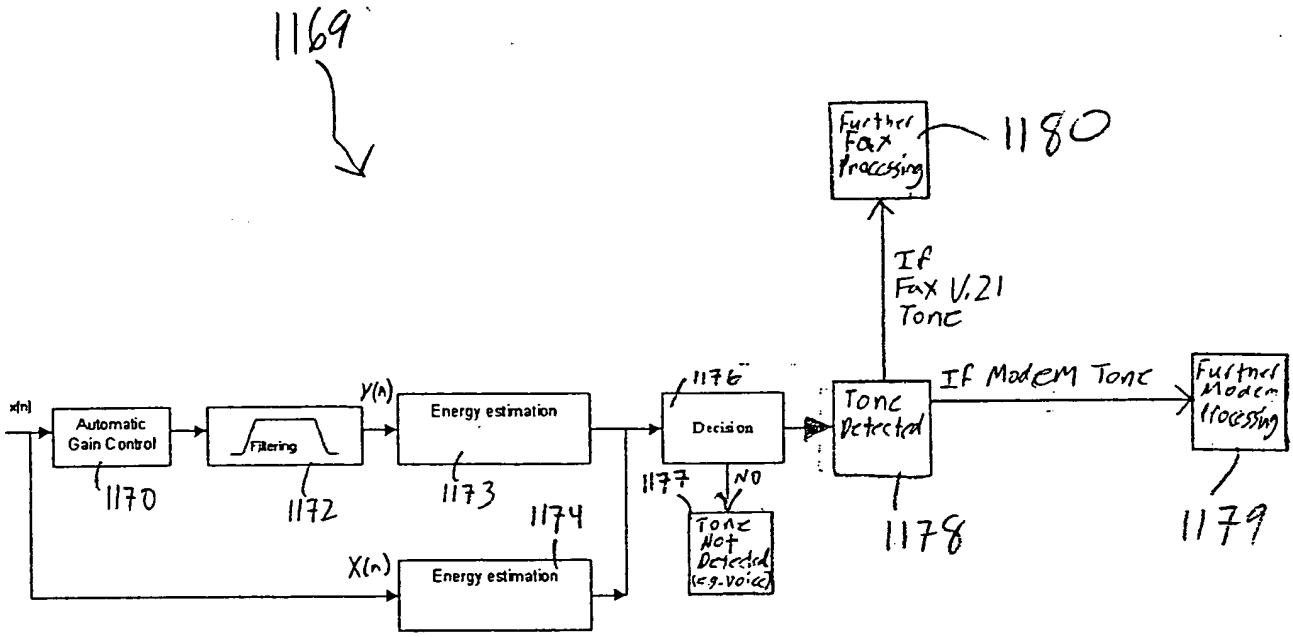

1169
Further Fax Processing
1180
If Fax V.21 Tone
If Modem Tone
Further Modem Processing
x[n]
Automatic Gain Control
1170
Filtering
1172
y(n)
Energy estimation
1173
1176
Decision
1177
NO
Tone Not Detected (e.g. voice)
Tone Detected
1178
1179
1174
X(n)
Energy estimation

FIG. 11E

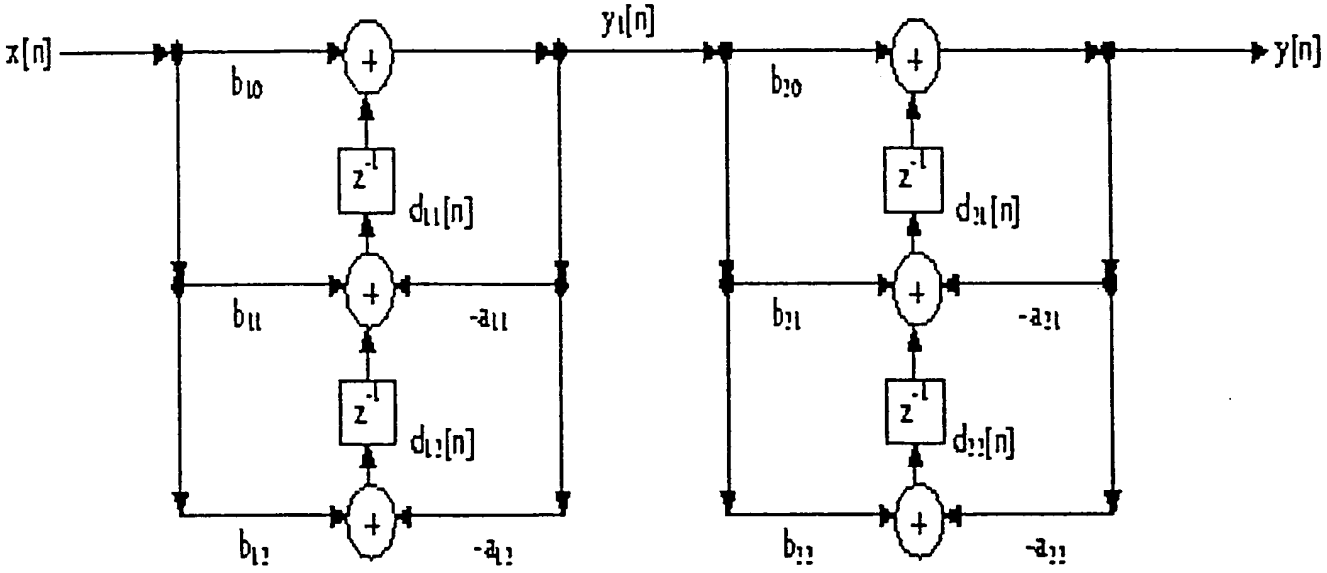
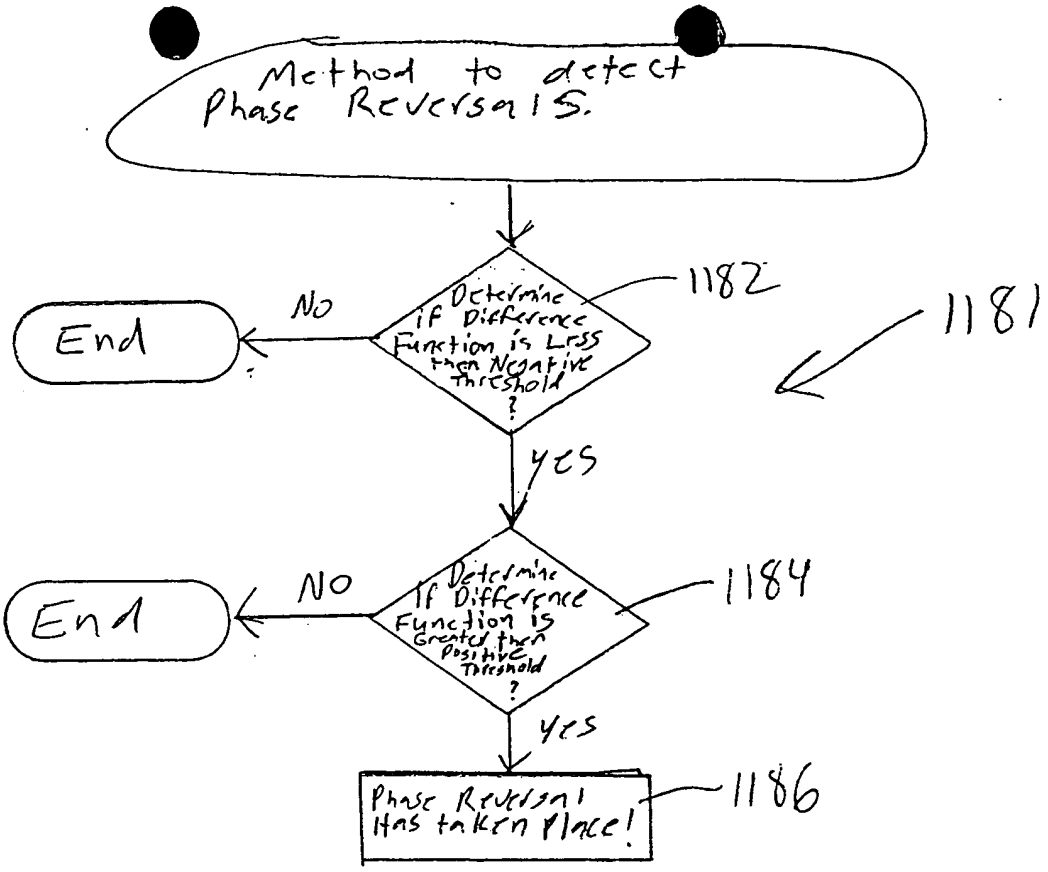

x[n]
y1[n]
y[n]
b10
b11
-a11
b12
-a12
d11[n]
d12[n]
b20
b21
-a21
b22
-a22
d21[n]
d22[n]
z-1
+

FIG. 11F


Method to detect Phase Reversals.
Determine if Difference Function is Less than Negative threshold ?
1182
1181
NO
End
yes
Determine if Difference Function is Greater than Positive Threshold ?
1184
NO
End
yes
Phase Reversal Has taken Place!
1186

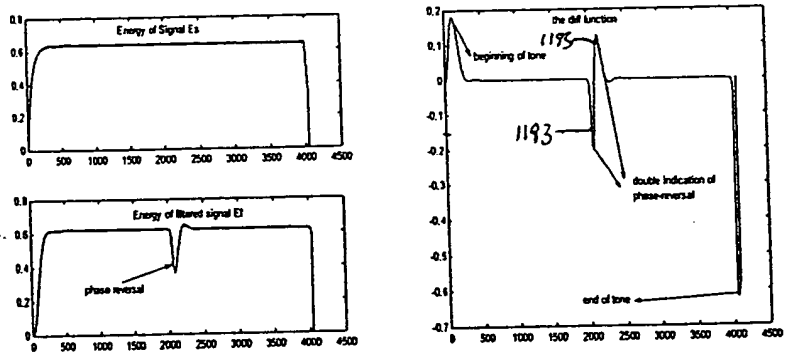

Energy of Signal Es
Energy of filtered signal Ef
phase reversal
the diff function
1195
beginning of tone
1193
double indication of phase-reversal
end of tone

FIG. 11G

Method for Fax V.21 Detection

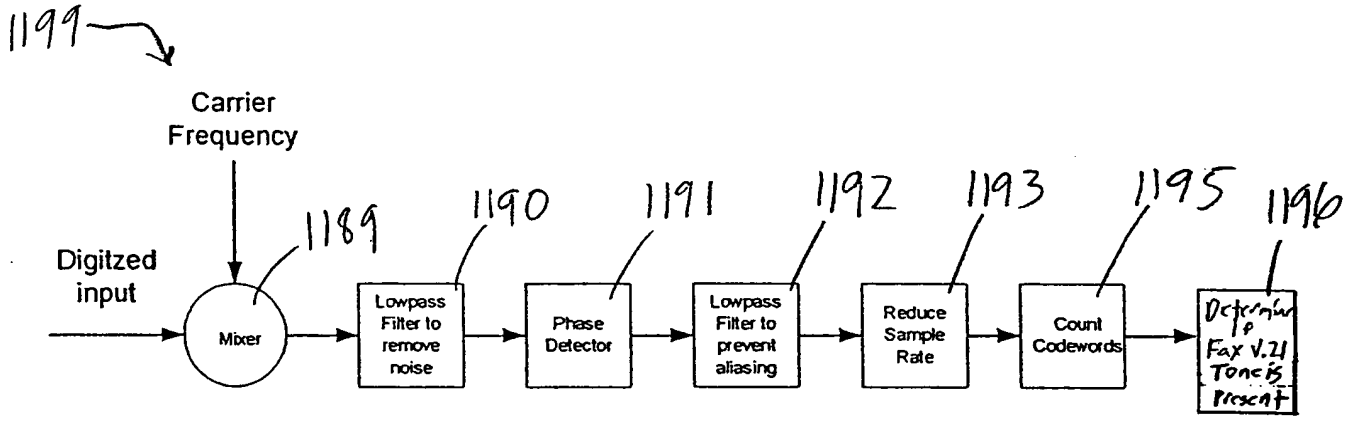



FIG. 11H

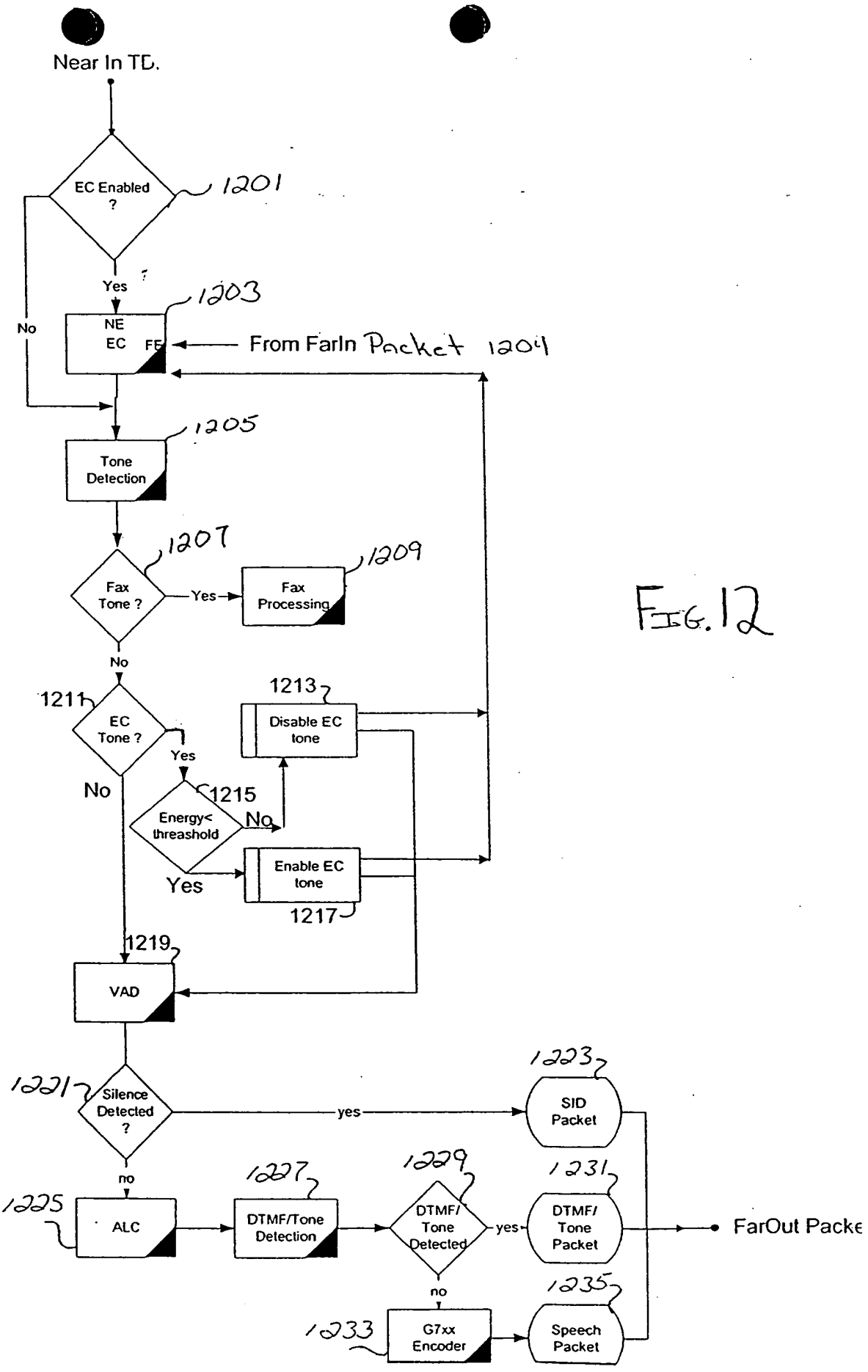

Near In TD.
EC Enabled ?
1201
Yes
No
1203
NE EC FE
From FarIn Packet 1204
1205
Tone Detection
1207
Fax Tone ?
Yes
Fax Processing
1209
No
1211
EC Tone ?
Yes
No
1213
Disable EC tone
1215
Energy< threashold
No
Yes
Enable EC tone
1217
1219
VAD
1221
Silence Detected ?
yes
1223
SID Packet
no
1225
ALC
1227
DTMF/Tone Detection
1229
DTMF/ Tone Detected
yes
1231
DTMF/ Tone Packet
FarOut Packe
no
1233
G7xx Encoder
1235
Speech Packet

FIG. 12

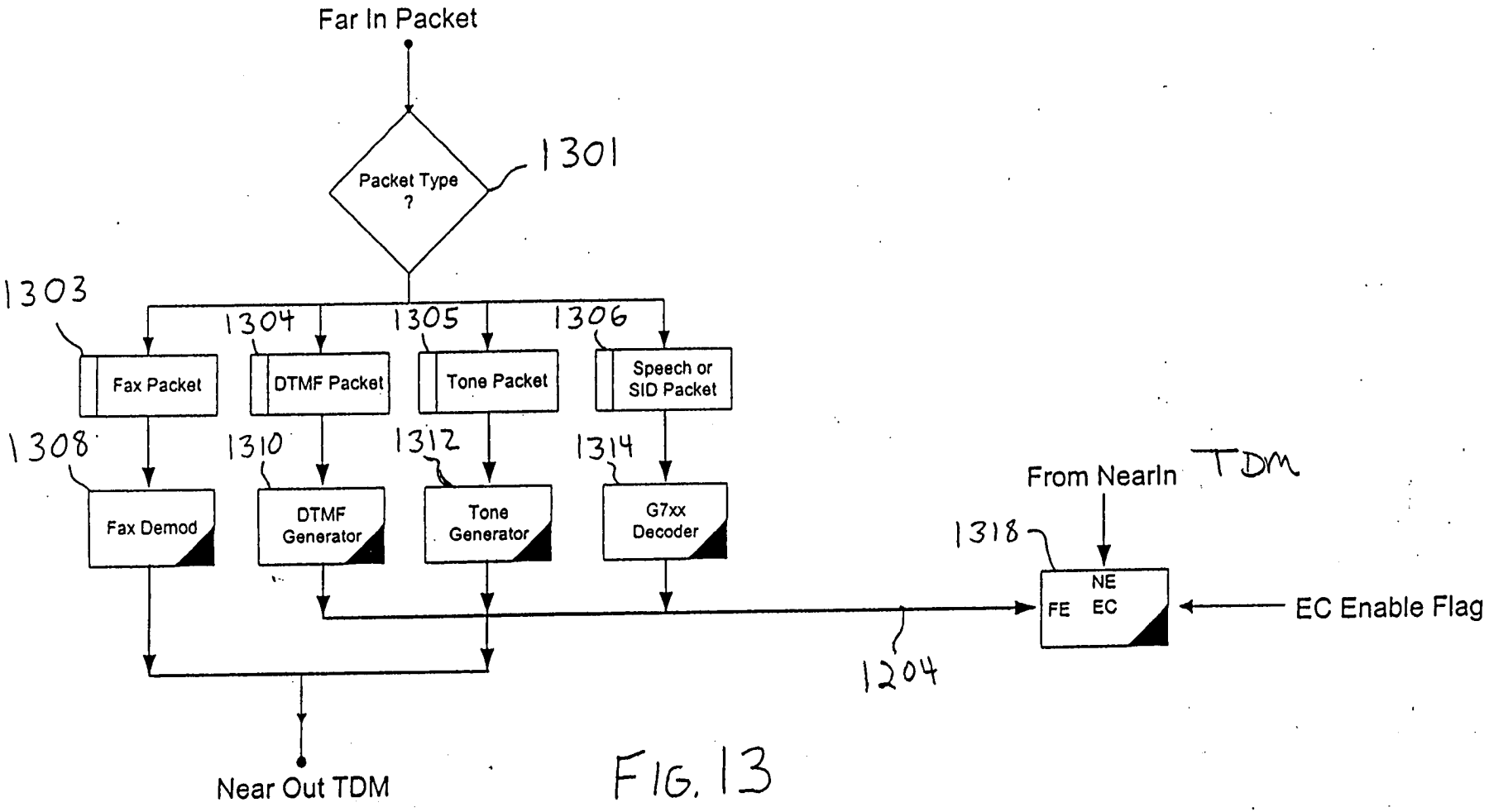

Far In Packet
Packet Type ?
1301
1303
1304
1305
1306
Fax Packet
DTMF Packet
Tone Packet
Speech or SID Packet
1308
1310
1312
1314
Fax Demod
DTMF Generator
Tone Generator
G7xx Decoder
From NearIn
TDM
1318
NE
FE
EC
EC Enable Flag
1204
Near Out TDM

FIG. 13

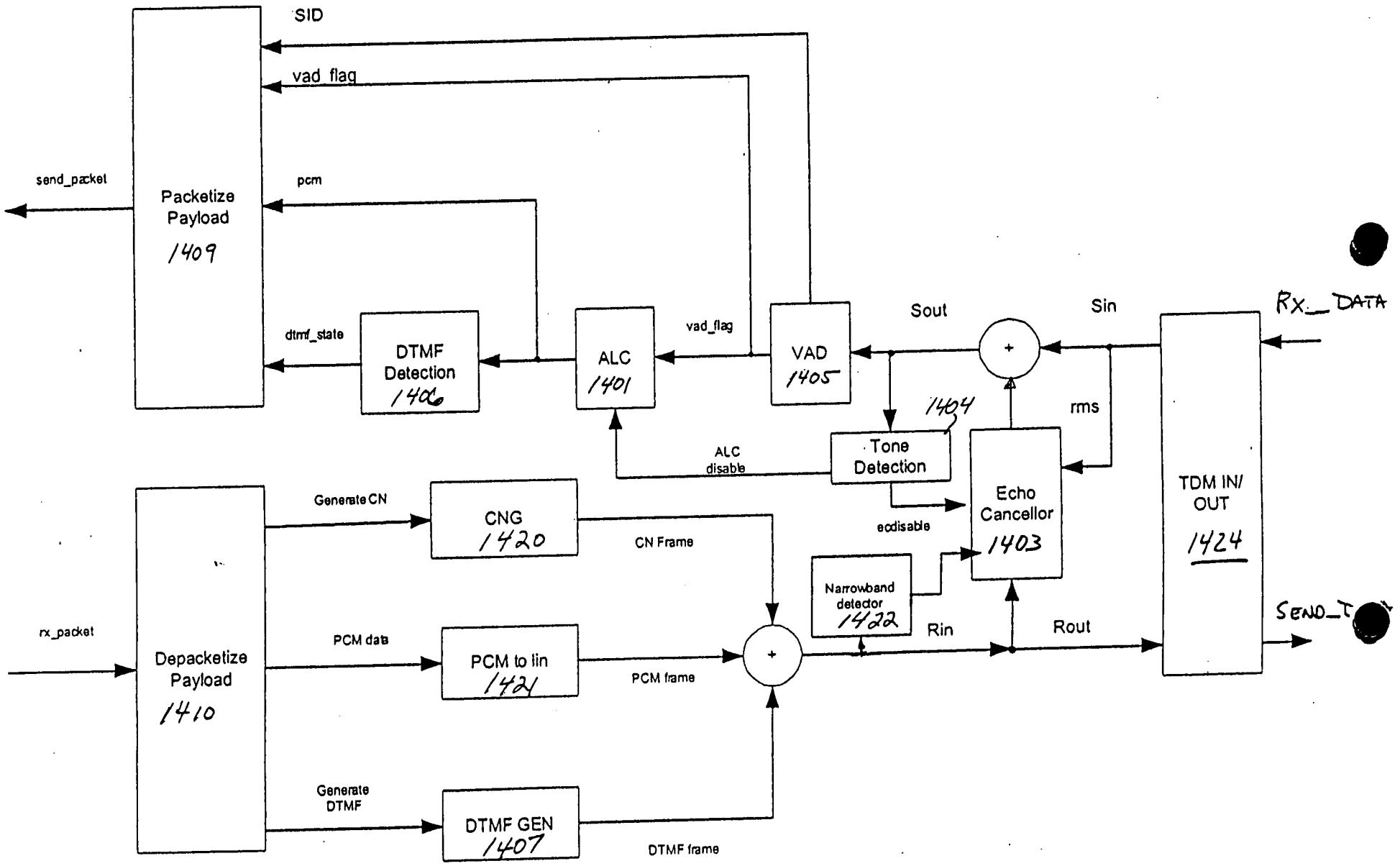

SID
vad_flag
send_packet
Packetize Payload
1409
pcm
dtmf_state
DTMF Detection
1406
ALC
1401
vad_flag
VAD
1405
Sout
Sin
RX_DATA
1404
rms
Tone Detection
ALC disable
Echo Cancellor
1403
TDM IN/ OUT
1424
ecdisable
Generate CN
CNG
1420
CN Frame
Narrowband detector
1422
SEND_D
rx_packet
Depacketize Payload
1410
PCM data
PCM to lin
1421
PCM frame
Rin
Rout
Generate DTMF
DTMF GEN
1407
DTMF frame

FIG. 14

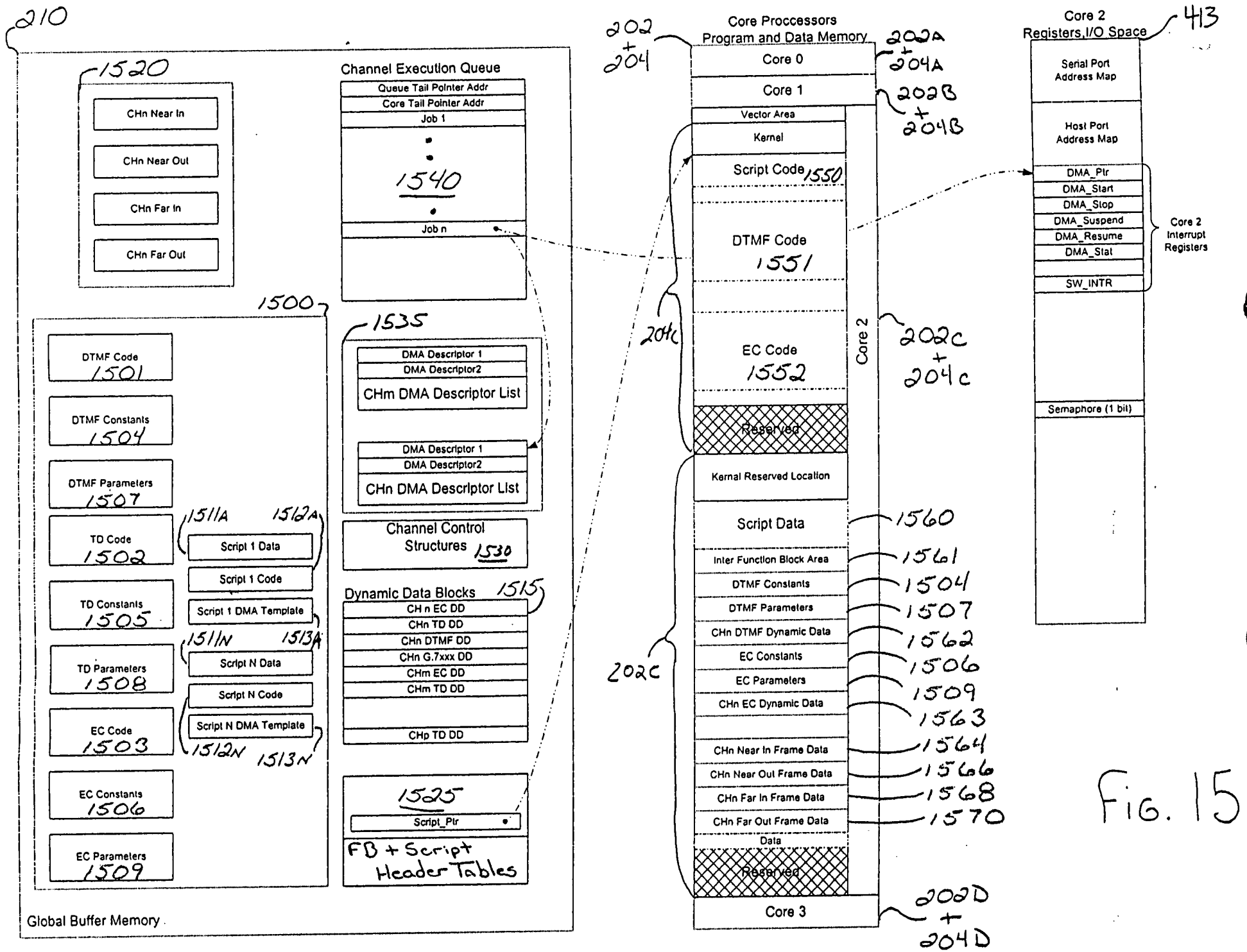

210
1520
CHn Near In
CHn Near Out
CHn Far In
CHn Far Out
Channel Execution Queue
Queue Tail Pointer Addr
Core Tail Pointer Addr
Job 1
1540
Job n
1500
1535
DMA Descriptor 1
DMA Descriptor2
CHm DMA Descriptor List
DMA Descriptor 1
DMA Descriptor2
CHn DMA Descriptor List
DTMF Code 1501
DTMF Constants 1504
DTMF Parameters 1507
TD Code 1502
TD Constants 1505
TD Parameters 1508
EC Code 1503
EC Constants 1506
EC Parameters 1509
1511A
1512A
Script 1 Data
Script 1 Code
Script 1 DMA Template
1511N
1513A
Script N Data
Script N Code
Script N DMA Template
1512N
1513N
Channel Control Structures 1530
Dynamic Data Blocks 1515
CH n EC DD
CHn TD DD
CHn DTMF DD
CHn G.7xxx DD
CHm EC DD
CHm TD DD
CHp TD DD
1525
Script_Ptr
FB + Script Header Tables
Global Buffer Memory
202 + 204
Core Proccessors Program and Data Memory
202A + 204A
Core 0
Core 1
202B + 204B
Vector Area
Kernel
Script Code 1550
DTMF Code 1551
EC Code 1552
Reserved
Kernal Reserved Location
Script Data
Inter Function Block Area
DTMF Constants
DTMF Parameters
CHn DTMF Dynamic Data
EC Constants
EC Parameters
CHn EC Dynamic Data
CHn Near In Frame Data
CHn Near Out Frame Data
CHn Far In Frame Data
CHn Far Out Frame Data
Data
Reserved
Core 3
Core 2
204C
202C + 204C
202C
1560
1561
1504
1507
1562
1506
1509
1563
1564
1566
1568
1570
202D + 204D
Core 2 Registers,I/O Space
413
Serial Port Address Map
Host Port Address Map
DMA_Ptr
DMA_Start
DMA_Stop
DMA_Suspend
DMA_Resume
DMA_Stat
SW_INTR
Core 2 Interrupt Registers
Semaphore (1 bit)

Fig. 15

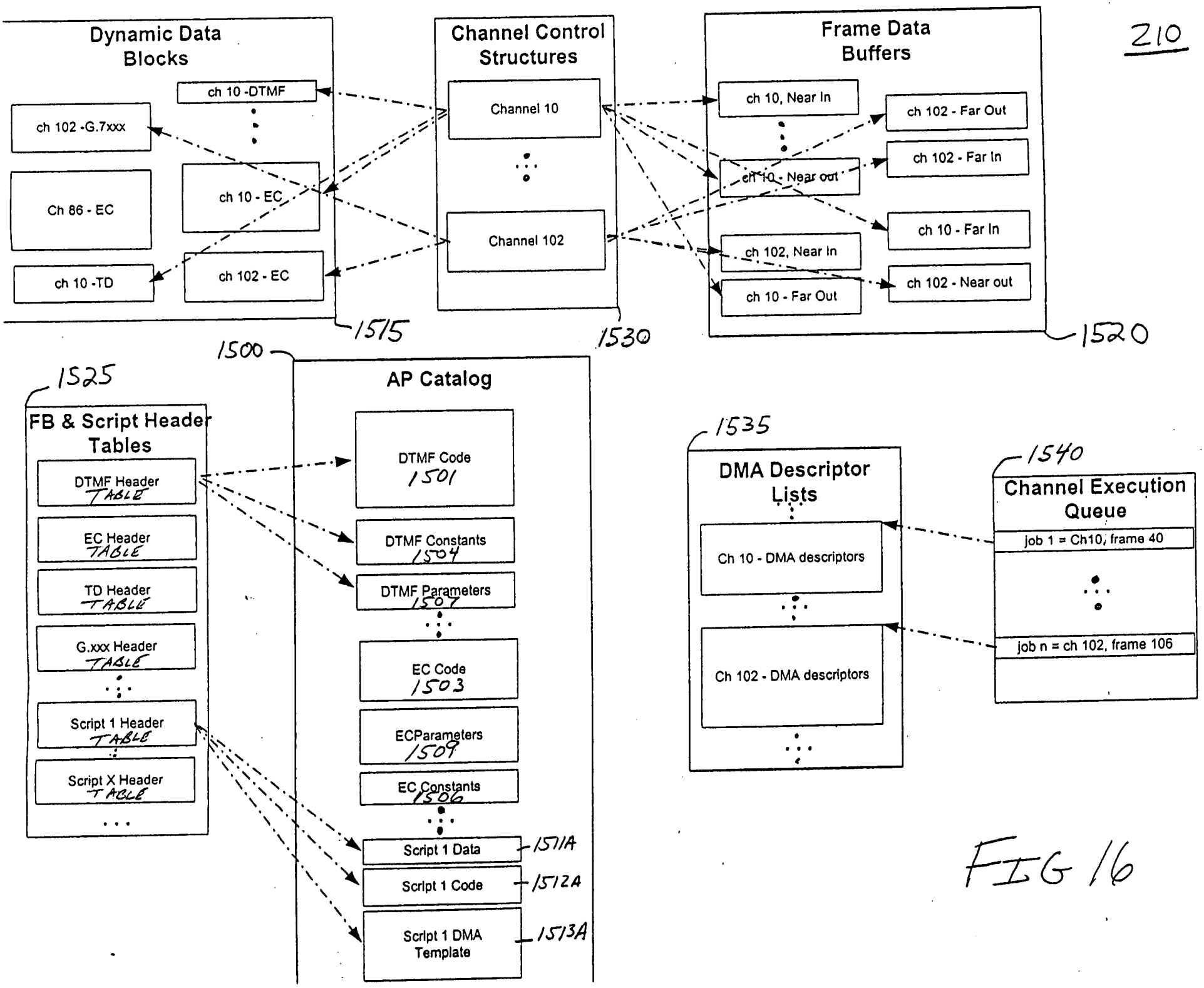

210
Dynamic Data Blocks
ch 10 -DTMF
ch 102 -G.7xxx
Ch 86 - EC
ch 10 - EC
ch 10 -TD
ch 102 - EC
1515
Channel Control Structures
Channel 10
Channel 102
1530
Frame Data Buffers
ch 10, Near In
ch 10 - Near out
ch 102, Near In
ch 10 - Far Out
ch 102 - Far Out
ch 102 - Far In
ch 10 - Far In
ch 102 - Near out
1520
1500
AP Catalog
1525
FB & Script Header Tables
DTMF Header TABLE
EC Header TABLE
TD Header TABLE
G.xxx Header TABLE
Script 1 Header TABLE
Script X Header TABLE
DTMF Code 1501
DTMF Constants 1504
DTMF Parameters 1507
EC Code 1503
ECParameters 1509
EC Constants 1506
Script 1 Data
1511A
Script 1 Code
1512A
Script 1 DMA Template
1513A
1535
DMA Descriptor Lists
Ch 10 - DMA descriptors
Ch 102 - DMA descriptors
1540
Channel Execution Queue
job 1 = Ch10, frame 40
job n = ch 102, frame 106

FIG 16

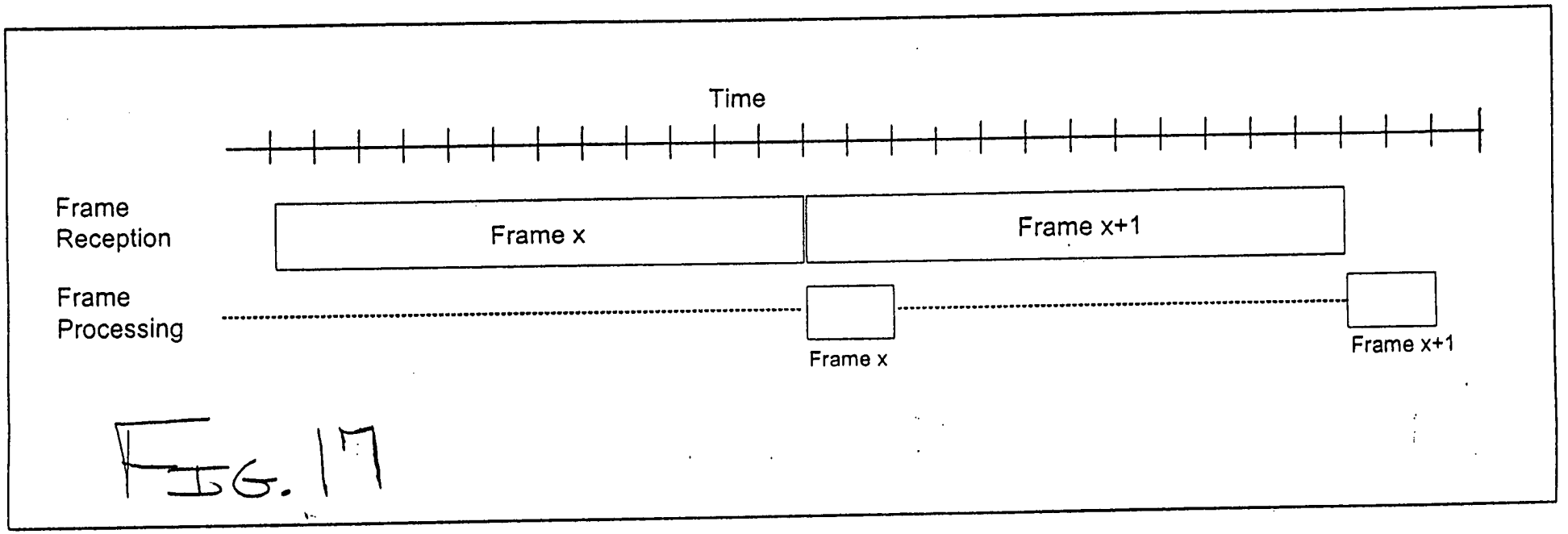
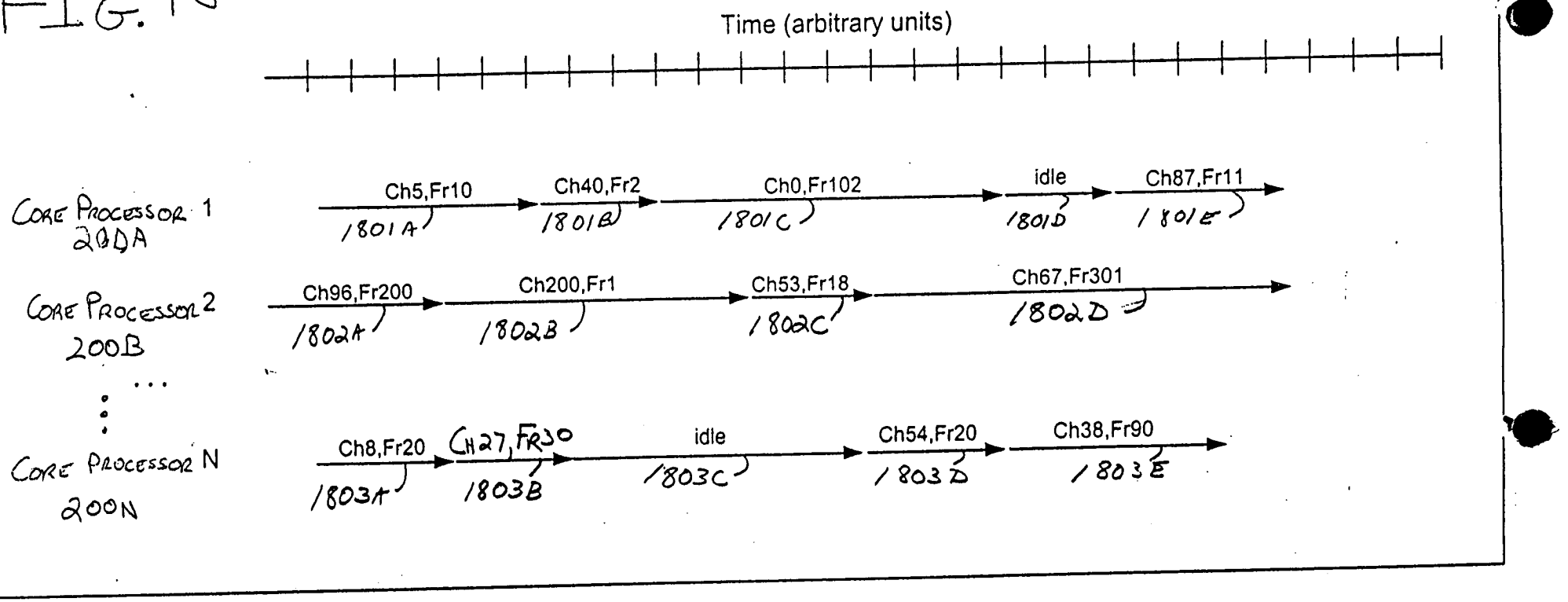

Time
Frame Reception
Frame x
Frame x+1
Frame Processing
Frame x
Frame x+1

FIG. 17

FIG. 18


Time (arbitrary units)
CORE PROCESSOR 1
200A
Ch5,Fr10
1801A
Ch40,Fr2
1801B
Ch0,Fr102
1801C
idle
1801D
Ch87,Fr11
1801E
CORE PROCESSOR 2
200B
Ch96,Fr200
1802A
Ch200,Fr1
1802B
Ch53,Fr18
1802C
Ch67,Fr301
1802D
CORE PROCESSOR N
200N
Ch8,Fr20
1803A
Ch27,Fr30
1803B
idle
1803C
Ch54,Fr20
1803D
Ch38,Fr90
1803E